THE UNIVERSITY of EDINBURGH

Edinburgh Research Explorer

Interactive Task Learning via Embodied Corrective Feedback

Citation for published version:
Appelgren, M & Lascarides, A 2020, 'Interactive Task Learning via Embodied Corrective Feedback', *Autonomous Agents and Multi-Agent Systems*, vol. 34, 54. https://doi.org/10.1007/s10458-020-09481-8

Digital Object Identifier (DOI):
10.1007/s10458-020-09481-8

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Publisher's PDF, also known as Version of record

Published In:
Autonomous Agents and Multi-Agent Systems

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 08. May. 2023

Autonomous Agents and Multi-Agent Systems (2020) 34:54
https://doi.org/10.1007/s10458-020-09481-8

Interactive task learning via embodied corrective feedback

Mattias Appelgren[1] · **Alex Lascarides**[1]

© The Author(s) 2020

Abstract
This paper addresses a task in Interactive Task Learning (Laird et al. IEEE Intell Syst 32:6–21, 2017). The agent must learn to build towers which are constrained by rules, and whenever the agent performs an action which violates a rule the teacher provides verbal corrective feedback: e.g. "No, red blocks should be on blue blocks". The agent must learn to build rule compliant towers from these corrections and the context in which they were given. The agent is not only ignorant of the rules at the start of the learning process, but it also has a deficient domain model, which lacks the concepts in which the rules are expressed. Therefore an agent that takes advantage of the linguistic evidence must learn the denotations of neologisms and adapt its conceptualisation of the planning domain to incorporate those denotations. We show that by incorporating constraints on interpretation that are imposed by *discourse coherence* into the models for learning (Hobbs in On the coherence and structure of discourse, Stanford University, Stanford, 1985; Asher et al. in Logics of conversation, Cambridge University Press, Cambridge, 2003), an agent which utilizes linguistic evidence outperforms a strong baseline which does not.

Keywords Human robot interaction · Interactive learning · Knowledge representation and reasoning

1 Introduction

The nascent field of Interactive Task Learning (ITL) aims to develop agents that can learn arbitrary new tasks through a combination of their own actions in the environment and an ongoing interaction with a teacher (see Laird et al. [41] for a recent survey). A current assumption for many AI systems is that any capabilities required can be programmed and trained prior to deployment. However, this assumption may be untenable for tasks that contain a vast array of contingencies. It is also problematic if the task is one where unforeseen changes to what constitutes successful behaviour can occur after an agent is deployed: for

✉ Mattias Appelgren
mattias.appelgren@ed.ac.uk

Alex Lascarides
alex@inf.ed.ac.uk

[1] School of Informatics, University of Edinburgh, 10 Crichton Street, Edinburgh EH8 9AB, Scotland, UK

Published online: 27 September 2020

instance, tasks where the set of possible options, or the specifications that govern correct behaviour, can change at any given time. Motivated by such issues, ITL seeks to create agents that can learn after they are deployed, through situated interactions which are natural to the human domain expert that they interact with.

Although interaction can take many forms, such as demonstration through imitation or teleoperation [6], our interest lies in approaches that make use of natural language to teach agents. A common formulation of such a learning process is as a situated and extended discourse between teacher and agent, much like one between a teacher and apprentice. During this discourse, the teacher provides instructions [55], describes current states [33] and defines concepts [53], goals [38], and actions [55], while the agent asks clarifying questions [54] and executes the instructed commands. Existing ITL systems that adopt these approaches tend to assume that the teacher's utterances are unambiguous, and that any uncertainty or lack of knowledge can be detected by the agent and rectified through asking relevant questions (e.g. [54, 55]). There is also an assumption that the teacher's guidance is geared towards what the agent should do now, rather than what the agent just got wrong. This emphasis effectively places a burden on teachers to recall and provide all the information that's required to perform the next appropriate action *before* the teacher observes the agent's attempts to do it. To alleviate or recover from that burden, it would be useful to support, and learn from, a dialogue move where the teacher expresses why the agent's latest action was suboptimal, or incorrect.

The aim of this paper is to fill this gap in ITL. In our task, the agent starts out with only partial knowledge of the goal that is a part of its planning problem. The true goal is for all blocks to be placed in a tower (the agent knows this) but there are additional constraints which place requirements on the final tower (e.g., each red block must be on a blue block), and the agent is ignorant of these additional constraints. It is also unaware of the domain level concepts (in our case, colours) that define them, and the natural language words that refer to them. We assume the teacher was unable to provide the agent with an exhaustive description of these constraints prior to deployment—due, for example, to issues with recall, unforeseen changes to the requirements or environment, etc. This assumption creates a natural context for a teacher to utter a particular kind of speech act, namely *correction* (e.g., "No, red blocks should be on blue blocks"). As we mentioned, this type of speech act—where the teacher expresses a specification of correct behaviour that has just been violated—hasn't been well studied in an ITL setting before. Our aim is to show that corrections are a useful source of evidence in ITL, in contexts where the agent needs to learn the goal that is a part of its planning problem, as well as how to ground concepts.[1]

The communicative intention of a correction is to change the beliefs or intentions of the corrected party [8]. In a purely verbal dialogue, a correction rejects parts of a previous utterance and (optionally) introduces alternative content, perceived by the corrector to be more accurate than the content of the corrected utterance. For example, in the simple dialogue below, utterance (1b) rejects who ate the last sandwich, but not that it was eaten:

[1] ITL has been used to learn novel actions, novel concepts that define the hypothesis space of possible states, and optimal policies. Here, we restrict attention to only the latter two learning tasks; we do not address using corrections to learn novel actions or motor skills.

(1) a. "Mary ate the last sandwich"
b. "No, it was Tom"

In the corrections we cover in this paper, the correction denies not aspects of a prior *verbal* utterance, but rather aspects of the latest physical action that's been performed by the agent. Specifically, it is rejecting that action from being part of a valid plan, given the constraints that define the planning problem. The dialogue move we focus on in this paper provides an explanation of why the action was rejected, by stating which constraint (or specification) was violated. Specifications that govern commercial processes can be conventions that appear quite arbitrary, rather than justified or motivated from commonsense knowledge. Specifications of the latter kind include those concerning health and safety, but there are many of the former kind that are designed simply to ensure consistency in the way people perform complex tasks on a production line (e.g., Amazon has a specification that packages containing batteries must have the package label on the left). So this paper ignores the task of learning specifications via commonsense knowledge and inference. It also doesn't address dialogue moves where the teacher justifies or explains why a particular specification exists. We focus instead on the agent learning specifications or constraints that appear arbitrary—in our case, they are expressed in terms of colour—and we address how the agent exploits its (uncertain) beliefs about specifications and how they apply in the current domain state during planning.

Previous work on task learning using correction has covered only simple "yes/no" feedback [39] or one word feedback paired with demonstrations [51]. This previous work shows the value of providing corrective feedback to the agent's latest action. Our contribution is to expand the scope of such moves, to consider utterances where the feedback contains actual linguistic content that describes the source of the agent's error, but which requires the acquisition of grounded language understanding to interpret. Dealing with the grounding problem in the context of correction introduces challenges which previous grounding work has not needed to contend with. For speech acts such as instruction and description, the content of the utterance is always a partial description of the non-verbal context. More technically, the non-verbal state satisfies the truth conditions of the speaker's utterance (or, in the case of instructions is a valid initial state for carrying out the described action). But this isn't the case with what Asher et al. [8] call divergent speech acts, such as corrections. With these speech acts, as we observed above in (1), the content of the correction and the item it corrects are mutually inconsistent. For instance, when the teacher corrects an agent's latest action by saying "No, that should be a blue block", the agent should infer that the denotation of the demonstrative "that" is *not* a blue block. In other words, it generates a negative training exemplar for grounding blue, not a positive one. More generally, in learning to ground symbols from corrections, the agent must infer what part of its latest action is being denied, and generate appropriate training exemplars—both positive and negative—from that (for details, see Sect. 4).

2 Related work

Interactive Task learning (ITL) aims to exploit interaction to support autonomous decision making during planning [41]. Similar to Kirk and Laird [38], our aim is to provide the agent with information about goals and concepts that allow it to construct a formal

representation of the decision problem, which standard decision making algorithms can then solve. The teacher does not provide a specific plan (as in e.g., Nicolescu and Mataric [50], She et al. [55]), but rather the information needed to infer a valid plan in a large range of contingencies. In our case we focus on learning goals which could arise from any number of sources, not just common sense knowledge or health and safety, but also apparently arbitrary regulations that enforce consistency across those performing the same task. The learned constraints are added to the goal description in PDDL, although it is conceivable that the learned constraints could be added in other ways, e.g. as done in Answer Set Programming for common sense rules Erdem et al. [19, 20]. We extend previous work in ITL by supporting a novel but natural way in which to express this information, namely *corrective* dialogue moves that highlight an aspect of the goal that's *violated* by the agent's latest action.

Corrections change the nature of the inferences an agent must make when learning to solve its task. In prior ITL tasks, utterances are usually asserted in a context where their intended contents are satisfied (e.g., [38]). Most approaches assume the teacher's utterance is either a *request* to perform a specific action (e.g., [3]), or it *describes* the current state (e.g., [42, 56, 57, 60]), or both are supported [48]. This means the non-linguistic context provides a positive example for learning to interpret the teacher's assertion: for instance, the agent can infer from the instruction "Put a red block on a blue block" that there is a red block and a blue block in the current visual scene that satisfy the preconditions of the put action, and its task in updating its model of symbol grounding is to estimate those positive exemplars from that scene, and update its grounding parameters with that positive evidence. However, for *corrections*, the current state is a situation where it is impossible to achieve the teacher's expressed goal without reversing the most recent action. For example, if the corrective move is "No, pick up only blue blocks", then the agent must infer that the block it just picked up is not blue—i.e., it's a negative examplar for grounding the word "blue". Inferring what aspect of the context caused the goal to be violated can be complex (see Sect. 4). But it's triggered by a very natural dialogue strategy for teachers to use—i.e., correcting a learner by expressing what caused the latest action to be a mistake. Developing methods that benefit from this type of evidence is the main aim of this paper.

Using natural language (NL) is a natural way for humans to communicate. It can provide evidence in a more data efficient manner than non-verbal demonstration alone: even simple yes/no feedback can be used to learn a reward function [39] or to trigger specific types of learning algorithms for correcting bad behaviour [51]. But dealing with more extended NL phrases raises additional challenges. First, it is necessary to map NL strings into formal semantic representations or logical forms that support inference (e.g., [63, 67]). As in prior ITL systems, (e.g., [22, 45]) we assume our agent begins with the ability to analyse the syntax of a sentence, but syntax doesn't resolve semantic scope ambiguities or lexical senses [13]. So while linguistic syntax restricts the possible logical forms to a finite set, the agent must use the context of utterance to identify which logical form matches the speaker's intended meaning.

Thanks to its knowledge of linguistic syntax and semantic compositionality, the agent is able to parse unforeseen words—i.e. neologisms—into predicate symbols of the correct arity. However, mapping NL utterances to semantic representations isn't sufficient for ITL. The agent must also learn to recognize the *denotations* of any newly added predicate symbols. In an embodied setting, this means estimating the set of percept values associated with those denotations. In other words, a mapping between the continuous percept space and discrete symbols must be found. This is the *symbol grounding problem* [16, 27]. Within artificial systems it has also been called the anchoring problem

[15], although anchoring takes into account additional difficulties, such as maintaining an estimate of an object's position, even when it becomes occluded by other objects. In this paper we deal only with the basic problem of mapping visual features to specific concepts—all objects in the scene remain completely visible at all times to our agent.

There are two main approaches to solving the grounding problem. In the first, the connection between language and vision is made through latent structures, usually neural models, without observing or explicitly inferring connections between individual words in the NL utterance and structures within the other modality, such as specific objects (e.g. [49, 56, 57]). The second approach is to build explicit classifiers for individual words and concepts, which are then combined with methods for joining these classifiers together to provide meanings of extended linguistic expressions via principles of semantic compositionality (e.g., [22, 40, 42, 48, 65]). This second approach has been the traditional choice in developing robot systems that learn to interpret instructions [17, 22, 40] (although see also Karamcheti et al. [36], Al-Omari et al. [2], Anderson et al. [3]). In this paper, we adopt the second approach.

Any number of different models can be used to perform this symbol grounding, such as SVMs [59, 66], Nearest Neighbor classifiers [17], and Deep Neural Models [34, 62]. New concepts can also be described in terms of previously grounded words—e.g. "a medicine box is a white box with a cross on it" [53]. We do not propose a new approach to symbol grounding *per se*. Rather, our interest lies in modelling how symbol grounding can be acquired from interactions. Specifically, we focus on scenarios where grounding is updated online from human-robot interaction, rather than batch-learned on a pre-defined dataset. Previous methods all focus on learning from interactions which somehow define or label particular objects in the scene through, for example, describing an action [33], pointing and providing a label [9, 18, 23], or playing I Spy [60]. Our novelty lies in introducing corrections and modelling their *interaction* between symbol grounding and the semantic constraints imposed by *discourse coherence*—that is, that the speaker's message must be in a specific *coherence relation*, such as *correction*, to some part of the context [8, 28, 47] (see Sects. 4 and 5 for how we use discourse coherence to restrict the interpretation of the language). In fact, our methods should work for any symbol grounding model with the following properties: (1) the output of the model is a probability distribution; (2) the model can be updated incrementally, as and when new evidence is observed; and (3) it supports supervised learning that is both noisy (i.e. some labels in the training data may be incorrect) and weak (i.e., it copes with latent mappings from natural language words to the region in the visual scene where its denotation is located).

In this paper we will assume that dialogue management and primitive motor skills are fixed and known in advance, and we focus on learning complex plans. The learner must update its beliefs about the goal of its planning problem and how its visual percepts reveal what's true of the current domain state, based on the evidence provided by its own actions and interactions with the teacher. It then uses established planning algorithms to infer a valid plan, given those beliefs about the goal, and the current domain state given its visual percepts. Our approach is novel compared with prior work in ITL in that we exploit formal semantic models of *discourse coherence* [29, 37, 43] to inform learning. We show how the semantics of discourse coherence can be leveraged to support learning for a type of dialogue move that has so far been ignored in ITL: the teacher corrects the agent's latest action by specifying the constraint that it violates.

3 The task

The task takes place in a blocks world where each planning problem instance consists of 10 coloured blocks that must be placed into 1-3 towers. The agent has access to a PDDL domain definition which describes the action *put*(x, y) which places object x on object y and *unstack*(x, y) which removes x from y and places x back on the table. Prior to interaction and learning, the agent also has access to a PDDL problem description as shown in Fig. 7: it includes an initial state in which 10 individual objects are clear and on the table, and a goal that all blocks must be in a tower. But while both these representations of the initial state and goal are satisfied in the ground truth planning problem, critical information is missing: the true goal includes further constraints (e.g., that each red block must be on a blue block); and the initial state not only lacks information about which blocks are which colours, but perhaps more fundamentally, the predicate symbols corresponding to colour terms are not a part of the agent's vocabulary for defining the set of possible states at all. Instead, a file is included which contains the RGB value of each object in the problem description. In our simulations, these RGB values were randomly generated for each object from a Gaussian distribution with mean and variance chosen to generate colours which match a particular concept (see Sect. 6 for details),[2] and the agent can observe these RGB values for each object, henceforth referred to as $F(x)$. The domain and problem is written in PDDL for the MetricFF planner and conforms to STRIPS with the addition of quantified goals and numeric fluents as defined in PDDL 2 and supported by MetricFF, as shown in Fig. 7.

Solving planning problems in general is PSPACE-complete Bylander [11] while blocks world problems are NP-complete Gupta and Nau [26]. Planning with the FF planner is NP-complete in general but in situations where there are no deadends (i.e. states from which there is no valid plan to the goal) it is $O(n^3)$ Hoffmann and Nebel [32].

As we mentioned earlier, the true planning problem, which the agent must solve, includes additional constraints, expressed as quantified statements in the goal description. The agent starts out ignorant of these constraints. Further, the (colour) predicate symbols that feature in the constraints and their denotations are respectively not a part of the agent's vocabulary for defining the planning problem or a part of its domain model. These constraints specify conditions that must be true of the towers once completed, as described in Table 1 and Eqs. 1–3. Generalising from the example rules in Table 1 to rules involving any colour terms and numbers, we consider rules of three forms:

$$r_1^{c_1,c_2} = \forall x \cdot c_1(x) \rightarrow \exists y \cdot c_2(y) \wedge on(x, y)^3 \tag{1}$$

$$r_2^{c_1,c_2} = \forall y \cdot c_2(y) \rightarrow \exists x \cdot c_1(x) \wedge on(x, y)^4 \tag{2}$$

$$r_3^{c,n} = \forall t \cdot (tower(t) \rightarrow count(c, t) \leq n)^5 \tag{3}$$

where c, c_1 and c_2 are colours (e.g., red, blue, maroon) and n is an integer—specifically $n \in \{1, 2, 3\}$. In words, r_1 and r_2 constrain the colours of blocks that are in an *on* relation:

[2] The colours are generated from an HSV space where mean and variance in the Hue dimension have been selected for concepts such as red and green with Saturation and Value mean and variance remaining constant for all colour concepts

for instance, $r_1^{red,blue}$ means that every red block must be on a blue block, while $r_2^{red,blue}$ means that every blue block must have a red block on it. Constraints of the form r_3 limit the number of blocks of a particular colour c in all towers to at most n: for instance, $r_3^{red,3}$ means that the number of red blocks in any tower t must be less than or equal to 3. The rules constrain the final tower. However, due to their nature and the nature of the available actions, if one of them is violated by a *put* action it will remain violated unless that *put* action is undone by an *unstack* action.

A simulated teacher observes the agent attempting to build the towers. Every time the agent takes an action that breaks one or more of the rules in G (i.e., the action must be undone in any valid plan to achieve G), the teacher provides verbal feedback, expressed in natural language—e.g., "no, put red blocks on blue blocks" (for rules of the form r_1 and r_2) or "no, you can only have two red blocks in a tower" (for rules of the form r_3). The feedback serves to correct the agent's mistake, and it provides an explanation as to why the action was incorrect. However, the verbal component may be ambiguous between several rules (see Table 1 and Sect. 4 for details). Thus, the agent must disambiguate the teacher's intended message while simultaneously learning to ground new terms in the embodied environment—the latter task amounts to learning which RGB values are members of which colour concepts (see Fig. 1).

4 Dialogue coherence

The agent learns to solve its planning problem via the teacher's dialogue moves. To simplify matters we assume that the teacher is coherent (that is, a specific coherence relation connects her utterance to its context), sincere (i.e. she believes what she says) and competent (i.e. what she believes is true). In other words we assume that the teacher will always give well-timed feedback (ie, she utters a correction as soon as the agent makes a mistake) and the feedback will always be true. In this study we do not deal with possibility that the teacher's message may be fallible.

For the purpose of this study, the teacher and learner share an understanding of the teacher's dialogue strategy, which is the following: if the agent performs a put action a that must be undone (via unstack) in any valid plan for reaching the goal G, then the teacher will correct the learner's action by uttering one of the corrective utterances u described in Tables 1 and 2 (which u she utters in which embodied context is expanded on in this section). That u corrects a is written $Corr(a, u)$, and the agent observes this move.

When an utterance u is used to correct an action a, written $Corr(a, u)$, then according to Lascarides and Asher [44], the semantics of $Corr(a, u)$ entails the following: (a) the content of u is true (in other words, *Corr* is a right-veridical relation), and (b) the content of u negates some part of the corrected action a. Given our assumptions about speaker coherence, sincerity and competence, the semantic consequences of a speaker's dialogue moves are actually true. In other words, for the purposes of this study, all of the following are equivalent: content that's entailed by a dialogue move, content that the speaker intended to express, and content that's actually true. In our task, since the teacher is correcting the agent's latest action $a = put(o_1, o_2)$, any correction u of a conveys that a must be undone (via the unstack action) in any valid plan (thus denying that it is part of a correct plan). So even if u were simply the cue phrase "no", signalling that it's a correction move, $Corr(a, u)$ entails that there is some rule r, which is a part of the goal G, that cannot be satisfied by the

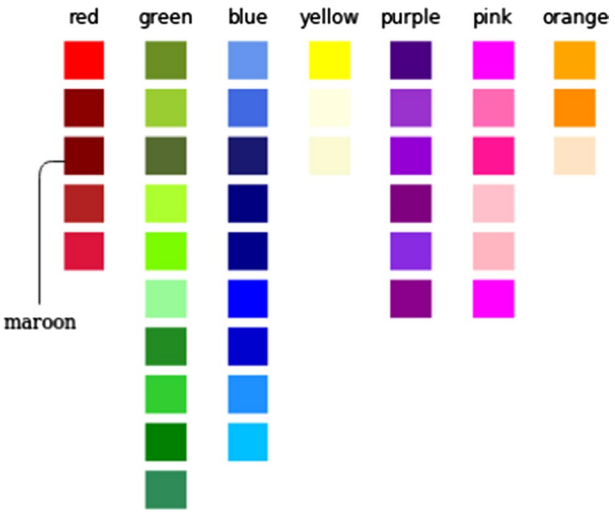


Fig. 1 The colours of objects fit into a number of different colour terms. Each individual instance of a colour is generated from a Gaussian distribution in HSV space, with mean and variance selected to produce colours described using a chosen category. The colour words used allow for high level categories such as "red" and "green" as well as more specific categories such as "maroon". This figure shows examples of hues generated in each category, including an example of a hue that fits both into the red and maroon categories (Color figure online)

Table 1 The teacher's natural language utterances, mapped to the set of possible constraints that they express

NL	Symbol	Logical form
Put red blocks on blue blocks	$r_1^{red,blue}$	$\forall x.red(x) \rightarrow \exists y.blue(y) \wedge on(x,y)$
	$r_2^{red,blue}$	$\forall y.blue(y) \rightarrow \exists x.red(x) \wedge on(x,y)$
Red blocks should be on blue blocks	$r_1^{red,blue}$	$\forall x.red(x) \rightarrow \exists y.blue(y) \wedge on(x,y)$
	$r_2^{red,blue}$	$\forall y.blue(y) \rightarrow \exists x.red(x) \wedge on(x,y)$
Put no more than 3 blue blocks in a tower	$r_3^{blue,3}$	$\forall t.(tower(t) \rightarrow count(blue,t) \leq 3)$

Specific colours and numbers can be replaced with other colours and numbers. The linguistic forms of two of the utterances do not fully determine the constraint the speaker intended to express. Conversely, it is possible to express a constraint in more than one way, although we only support a limited set of natural language surface forms in our experiments

outcome state s of the action a or any state s' that's reachable from s without first undoing a—in this case, we say that r is violated by a, written $V(r, a)$. We will define more precisely the various ways in which an action a can violate the rules r that feature in Table 1 shortly.

As we mentioned earlier, the teacher's correction moves u consist of more than the word "no". They also provide an *explanation* as to why the teacher said "no", by expressing which rule r was violated by the action: e.g., "No, red blocks should be on blue blocks". However, as shown in Table 1, the correcting utterance u is ambiguous as to which rule r the speaker intended to express: its linguistic form yields a set $M(u)$ of rules which could be meant by the utterance u, that set may consist of more than one rule, and which one of these the speaker actually intended to convey is hidden to the agent. So in our study, the corrective move $Corr(a, u)$ is satisfied if and only if one of the rules $r \in M(u)$ is both a part of the goal G and violated by a (meaning a must be undone in any plan to achieve r):

$$Corr(a,u) \leftrightarrow \bigvee_{r_i \in M(u)} (r_i \in G \wedge V(r_i,a)) \tag{4}$$

Equation (4) follows directly from the truth conditional semantics of correction and the assumption that the speaker is sincere, competent and coherent. The compositional semantics of u, combined with the speaker's sincerity and competence, means that there must be some rule $r \in M(u)$ that's true; and further, given our task, r being true is equivalent to $r \in G$. Further, if a doesn't violate the rule r, then the speaker failed to coherently explain why she used the word "no", to signal correction—for instance, a constraint that each red block should be on a blue block cannot coherently explain why putting a yellow block on a green block is a mistake, because the former isn't inconsistent with the latter. But which rule $r \in M(u)$ satisfies the conjunction in (4) is not determined by u's linguistic form alone: the agent must estimate it via the embodied context of utterance.

As shown in Table 1, the utterance u that the teacher says is the same if $r_1^{red,blue}$ is violated or if $r_2^{red,blue}$ is. In both cases the teacher will utter u_1 = "no, red blocks should be on blue blocks" (or its paraphrase, for this task at least, of "no, put red blocks on blue blocks"). If $r_3^{red,1}$ is violated, the teacher says u_2 = "no, you can only have one red block in any tower". Finally, in a situation where neither $r_1^{red,blue}$ (or $r_2^{red,blue}$) nor $r_3^{red,1}$ is violated in isolation, but their combination is violated, meaning that the most recent action must be undone to complete a valid tower, then the teacher says u_3 = "no, you can only have on red block in any tower, and red blocks must be on blue blocks" (see Sect. 5.2.7). We'll now discuss these in more detail, exploring how the truth conditions of coherent corrective moves, and in particular the conditions under which $V(a, r)$ is satisfied, impose constraints on the relationship between the speaker's communicative intent and the colours of various blocks in the visual scene. We do this in order to identify what the agent can infer when it observes one of the corrections u_1, u_2, or u_3.

4.1 Red blocks should be on blue blocks

When the teacher says "no, red blocks should be on blue blocks" this is ambiguous between the rules $r_1^{red,blue}$ and $r_2^{red,blue}$ (henceforth shortened to $r_1^{r,b}$ and $r_2^{r,b}$). The agent therefore needs to figure out which of the two rules was actually violated, in order to learn accurate information from the correction.[3]

In accordance with Eq. 4, an utterance is only used to correct an action if one of the rules which it could convey is both in the goal and is violated by that action. Therefore, since the possible messages of u_1 are $r_1^{r,b}$ and $r_2^{r,b}$, at least one of them must be in the goal and violated by the action a:

$$Corr(a,u_1) \leftrightarrow (r_1^{r,b} \in G \wedge V(r_1^{r,b},a)) \vee (r_2^{r,b} \in G \wedge V(r_2^{r,b},a)) \tag{5}$$

[3] It would be possible to resolve this ambiguity through prosodic stress [52]. With capitals indicating stress, "no, red blocks should be on BLUE blocks" resolves to $r_1^{r,b,}$ and "no, RED blocks should be on blue blocks" resolves to $r_2^{r,b}$. However, current automated speech recognizers do not reliably map prosodic stress to information about focus (and there is little consensus on how this should be done [10, 12]). Furthermore, ambiguity is an inherent in natural language and generally unavoidable, and so we purposefully ensure that the observed linguistic form leaves this ambiguity unresolved, so that we can study how it can be resolved by relating the utterance and its set of possible meanings to the context in which the utterance was made.

As mentioned earlier, saying u_1 = "No, red blocks should be on blue blocks" would be incoherent if the action a was somehow irrelevant to the possible messages $r_1^{r,b}$ and $r_2^{r,b}$, such as placing a green block on a yellow block.

Rules such as $r_1^{r,b}$ and $r_2^{r,b}$ can be violated in two different ways. First of all, in what we call a *D*irect violation, consider S_1 in Fig. 2. If $r_1^{r,b} \in G$, then an action resulting in S_1 would directly violate the rule since $r_1^{r,b}$ requires each red block to be on a blue block, but here a red block was put on a non-blue block. This *D*irect violation is expressed as (6), and similarly S_2 directly violates $r_2^{r,b}$ because of (7):

$$V_D(r_1^{r,b}, put(o_1)) \leftrightarrow (red(o_1) \wedge \neg blue(o_2) \wedge on(o_1, o_2)) \tag{6}$$

$$V_D(r_2^{r,b}, put(o_1)) \leftrightarrow (\neg red(o_1) \wedge blue(o_2) \wedge on(o_1, o_2)) \tag{7}$$

Rule $r_1^{r,b}$ is not directly violated in S_2 nor is $r_2^{r,b}$ directly violated in S_1. However, they illustrate the second kind of violation. These rules are respectively what we call *I*ndirectly violated: the towers themselves do not violate the rule, but it is now impossible to create rule-compliant towers, given the remaining blocks on the table, without first undoing that latest action. In S_2, $r_1^{r,b}$ is indirectly violated because by putting a non-red block on the blue one, the learner created a situation where there are more red blocks on the table than blue ones, and so one of those remaining red blocks has nowhere to go. S_1 indirectly violates $r_2^{r,b}$ for similar reasons. Indirectly violating $r_2^{r,b}$ can also happen if a places a red block in a tower after first placing a blue block in a second tower, resulting in a similar lack of red blocks to complete the tower with the blue block. These indirect violations of the rules are regimented as follows:

$$\begin{aligned} V_I(r_1^{r,b}, put(o_1, o_2)) \leftrightarrow (&\neg red(o_1) \wedge blue(o_2) \wedge on(o_1, o_2) \wedge \\ &|\{x : red(x) \wedge on(x, table)\}| > |\{y : blue(o_4) \wedge on(y, table)\}|) \end{aligned} \tag{8}$$

$$\begin{aligned} V_I(r_2^{r,b}, put(o_1, o_2)) \leftrightarrow (&red(o_1) \wedge \neg blue(o_2) \wedge on(o_1, o_2) \wedge \\ &(|\{x : blue(x) \wedge on(x, table)\}| > |\{y : red(y) \wedge on(y, table)\}|) \vee \\ &\exists x \exists t(blue(x) \wedge tower(t) \wedge top(t, x) \wedge \\ &|\{x : blue(x) \wedge on(x, table)\}| \geq \\ &|\{y : red(y) \wedge on(y, table)\}|)) \end{aligned} \tag{9}$$

When uttering u_1, our teacher helps the agent to determine which of these two types of violation has happened by pointing at the tower that the learner has just added a block to if it's a *D*irect violation V_D, and pointing at a block that can no longer be a part of a rule-compliant tower if it's an *I*ndirect violation V_I. So if $r_1^{r,b}$ is indirectly violated, then the teacher points at a red block on the table. If $r_2^{r,b}$ is indirectly violated, the teacher points to a blue block, either on the table or on the top of a tower, which no longer has a red block to be placed on top of it to conform with the rule.

If the agent can ground the colour terms "red", and/or "blue", then it could use the coherence Eqs. 6–9 to infer whether the teacher intended to convey $r_1^{r,b}$ or $r_2^{r,b}$. Conversely, if the agent knew which message the teacher intended to convey, then the agent could infer the colour of the relevant objects. However our agent may not know the intended message nor know how to ground the relevant colour terms. In this case we allow the agent to utter a query whose response will resolve the ambiguity. The above

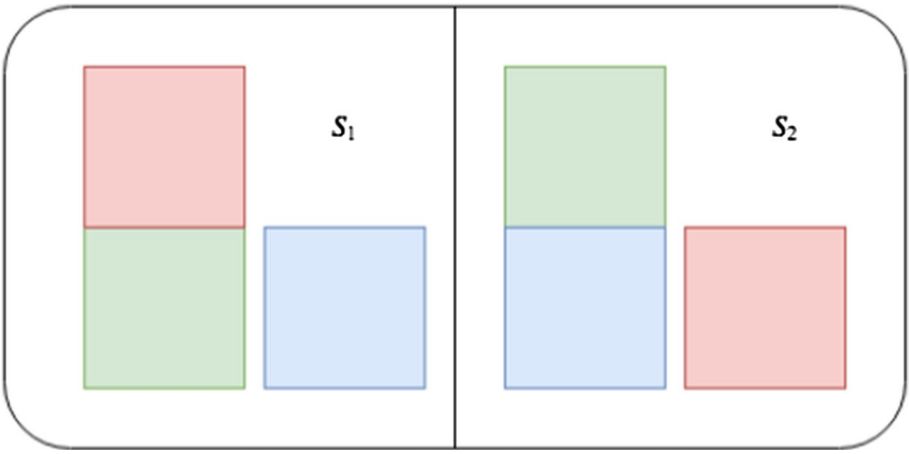


Fig. 2 Two states which violate $r_1^{red,blue}$ or $r_2^{red,blue}$ either directly or indirectly

coherence equations ensure that determining the colour of one of the available blocks would be sufficient to infer the correct interpretation; thus the agent asks a simple yes/no question about the colour of a block: for example, "is the block I just placed red?". By the above equations, the answer, whether "yes" or "no", is sufficient for the agent to infer both the speaker's intended message and the colours of the relevant blocks.

4.2 Goals with colour count rules of the form r_3

In an extension to the planning problems we considered in Appelgren and Lascarides [5], we now contemplate a further type of rule that can be a part of the goal G—see (3). Adding rules of this form creates situations where no individual rule in the goal G is violated by the latest action, indirectly or otherwise; however, *conjunctions* of rules in G are. In Sects. 5 and 6, we'll explore how the approach taken in Appelgren and Lascarides [5] scales to these more complex situations.

Specifically, consider corrective moves of the form u_2 = "No, you can only have one red block in any tower". Unlike u_1, there is only one rule in the set of possible messages of this utterance, namely $r_3^{r,1}$.[4] However, the challenge with adding rules of the form r_3 to G lies in the more complex ways in which they can be violated by the latest action, and how one can leverage information about grounding colour terms in spite of the added complexity.

There are two ways in which the rule $r_3^{r,1}$ can be violated by the latest action a. The first is when the learner puts a red block on top of a tower t that already has a red block in it; see S_3 in Fig. 3. This *D*irect violation is formalised in (10), where $a = put(o_1, o_2)$:

$$\begin{aligned} V_D(r_3^{r,1}, put(o_1, o_2)) \leftrightarrow\ & red(o_1) \wedge \exists x \exists t(tower(t) \wedge in(o_1, t) \wedge in(x, t) \wedge red(x) \\ & \wedge \forall y(in(y, t) \wedge x \neq y) \rightarrow \neg red(y)) \end{aligned} \tag{10}$$

[4] Utterance u_3 generates semantic scope ambiguities, but none of the alternative resolutions match any of the three rules we consider here.

In words, the top block o_1 is red, and exactly one of the blocks already in the tower is also red. This contrasts with direct violations of rules r_1 and r_2—these rules constrain the colours of only the top two blocks, rather than the colours of all blocks in the tower. However, the agent cannot tell in advance of the learning process whether utilizing the information in the second (existentially quantified) conjunct to learn symbol grounding is worthwhile (in that it improves data efficiency). Thus, whether it is worth utilizing a monotonic entailment of a dialogue move is an empirical matter. In Sect. 6, we'll present experiments that explore the trade off between exploiting monotonic entailments about speaker meaning and data efficiency in learning the task (see experiment 4).

The second way in which the action a may violate $r_3^{r,1}$ is when a doesn't create a tower with more than one red block; however, there is another rule in G which will be violated by the tower *unless* another red block is added to it. In other words, a can (indirectly) violate the *conjunction* of $r_3^{r,1}$ and another rule or the form $r_1^{r,x}$ or $r_2^{r,x}$ for some colour x, even though it doesn't violate either rule in isolation. For example, suppose that $r_2^{r,b}$ and $r_3^{r,1}$ are both a part of the goal G. Suppose, furthermore, that the agent has already put a red block in a tower, and now it puts a blue block on top of that tower. The agent cannot now satisfy both rules without first undoing a: either it now places a red block on top thereby satisfying $r_2^{r,b}$ but (directly) violating $r_3^{r,1}$; or it doesn't, in which case it (directly) violates $r_2^{r,b}$. The combination $r_3^{r,1} \wedge r_1^{r,b}$ can also be violated by this situation, if there are not enough blue blocks left to place any remaining red blocks on, thus forcing the agent to add a red block on top of the blue.

Gricean principles of cooperative conversation [25] predict that if the situation is one where the conjunction of two rules is not achievable from the current state, but either conjunct is, then the speaker should utter the conjunction as their corrective move, not just one of the conjuncts. This is because to mention only one conjuncts carries, via the Gricean maxims, a *scalar implicature* that this rule *alone* cannot be achieved from the current state, and in the situations we're considering this is misleading. Indeed, the above semantics for correcting actions with utterances u_1 and u_2 carry those scalar implicatures, following the formal semantics of correction in Lascarides and Asher [44]. Therefore, in order to reflect how human speakers reason about scalar implicatures when choosing what to say, the teacher uses the utterance u_2 when r_3 is violated in isolation, but expresses the conjunction of rules when the conjunction is violated but neither conjunct in isolation is: e.g., u_3 = "no, red blocks should be on blue blocks and you can only have one red block in each tower". Since the agent and teacher share the knowledge that this is the teacher's strategy (in essence, they are both following the Gricean maxims of conversation), the agent can make inferences about the colours of blocks from the teacher's signal. When the teacher utters u_2 or u_3, the agent can (monotonically) infer that the tower that the agent just added a block to already has a red block in it. Further if the teacher utters u_2, then the top block that the agent just added to the tower is red; if they utter u_3 it is blue.

However, the linguistic form of u_3 is *insufficient* to infer whether $r_1^{r,b}$ or $r_2^{r,b}$ is part of the goal G, for similar reasons that the linguistic form of u_1 is insufficient. Furthermore, unlike the situations in Sect. 4.1, the speaker doesn't use pointing to discriminate between a direct violation of r_1 (or r_2) and an indirect violation. In this context, it's not appropriate to point because by definition, the tower itself does not currently violate any rule, and pointing to a block on the table that cannot now be added in a goal-compliant manner fails to disambiguate the situation. To see this, consider the state S_5 in Fig. 4. Suppose that the teacher says u_3 ="No, put red blocks on blue blocks and there can be at most one red block in any given tower" in this state. Then even if the agent knows the colours of all the blocks, it is impossible to infer whether $r_3^{r,1} \wedge r_1^{r,b} \in G$ or $r_3^{r,1} \wedge r_2^{r,b} \in G$. The latter would be violated

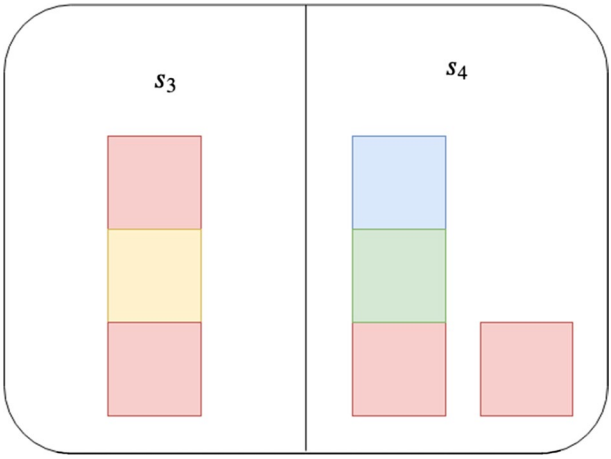


Fig. 3 Two alternative ways of violating $r3$: on its own ($S4$) or in combination with another 'red on blue'-rule (Color figure online)

Table 2 The natural language utterances used as examples in this section

Symbol	Utterance
u_1	Red blocks should be on blue blocks
u_2	Put no more than 3 blue blocks in a tower
u_3	Red blocks should be on blue blocks and put no more than 3 blue blocks in a tower
u_4	That is wrong for the same reason
u_5	That is not red either

by this state, because $r_2^{r,b}$ requires you to put a red block on top of the blue one, creating a tower with two red blocks thereby violating $r_3^{r,1}$. The former is also violated, because given the available blocks on the table, the only way to ensure that each red block is on a blue block is to add a red block to the tower, thereby (again) violating $r_3^{r,1}$. Notice that pointing to a red block on the table doesn't really help: in both cases this block cannot be added in a rule-compliant manner. On hearing u_3, the agent could attempt to make an informed decision about whether the rule violated is $r_1^{r,b}$ or $r_2^{r,b}$ by thinking about in what situations these distinct rules would necessitate a red block being placed on the blue block. If $r_2^{r,b}$ is a part of G, then a red block must be placed onto the tower *whatever* remains on the table. On the other hand, if $r_1^{r,b}$ is a part of G, then the teacher would have said u_3 only if there are more red blocks than blue ones remaining on the table. So counting the number of blue and red blocks potentially helps to resolve the ambiguity: if there are fewer or an equal number of reds than blues, then $r_1^{r,b}$ is violated; otherwise, either rule might be violated. Evaluating this inference in the visual scene is extremely challenging computationally: the learner must evaluate for each block in the scene whether it is red, blue, or neither. We believe this represents a poor trade off between performing the inference on the one hand and the value of the inferred information for estimating G and symbol grounding on the other. So we ignore it here. Nevertheless, our learning agent must disambiguate the (combination) of rules that are violated in this situation, even though it currently doesn't know the colours of the blocks. We'll show how this can be achieved in Sects. 5 and 6.

4.3 Using anaphora to resolve ambiguity

The utterances so far receive a coherent interpretation through their connection to the agent's latest action a. However, thanks to the presence of anaphoric expressions, some utterances can only be interpreted through their connection to a previous utterance in the dialogue. Previous work on anaphora have focused on referring expressions [21, 64]—in particular pronouns such as "it"—trying to identify particular objects in a scene from descriptions such as "in the cluster of orange objects at the bottom, it's the top object" [21]. In this work we focus on anaphoric expressions with richer linguistic content than pronouns: such expressions trigger *presuppositions* van der Sandt [61], whose context-specific interpretations are dependent on identifying how they coherently contribute to the prior discourse Asher and Lascarides [7].

Specifically, we consider two utterances with these properties: u_4 = "no, that is wrong for the same reason" and u_5 = "no, that is not red either" (or any other colour). Definite descriptions of the form "the same X", like in u_4, presuppose an antecedent in the context that satisfies the description X. The construction "not Y either", like in u_5, likewise presupposes an antecedent in the context that isn't Y. Thus utterance u_4 presupposes a prior (identical) reason (in our task, a rule violation) is a part of the multimodal context; u_5 presupposes that something else (in the context) is not red.

In line with existing coherence-based theories of discourse (eg., Hobbs [29], Kehler [37], Asher et al. [8]) we assume that any utterance containing an anaphoric expression or presupposition must be coherently connected to the unit that contains its antecedent. Thus u_4 (or u_5) must coherently attach to more than just the agent's latest action a; it must also attach to a prior utterance—this is why starting a dialogue with u_4 or u_5 sounds anomalous. Constraints on which parts of an *embodied* dialogue context the current utterance can coherently connect to are not yet fully understood (though see Hunter et al. [35] for initial work). We therefore take a very permissive approach: in principle, u_4 (or u_5) can coherently attach to any prior dialogue move. However, in line with all coherence-based theories of discourse interpretation, we adopt a preference for attachment to the most recent utterance u that supports a coherent interpretation, and in particular supplies the required antecedent. In other words, an utterance of the form u_4 or u_5 attaches with *correction* to the latest agent's action a, but also to the most recent prior utterance u where a coherence relation $R(u, u_4)$ (or $R(u, u_5)$) can be established and an antecedent in u to the anaphoric expressions ins u_4 (or u_5) identified.

The utterance u_4 can be interpreted as an *elaboration* of any prior correction u: even if u were simply the expression "no": thanks to (5) a violated rule r can be accommodated as part of the content of u precisely because it corrects an agent's (prior) action. Thus in embodied dialogue (2), (2f) attaches to the action (2e) with *correction* and also to (2d) with *elaboration* because (2d) is more recent than (2)b):

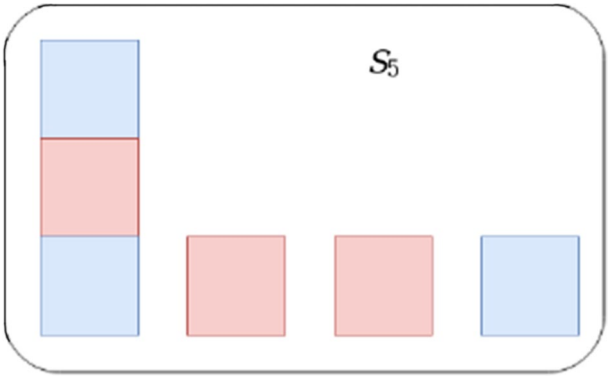


Fig. 4 A state in which "No, put red blocks on blue blocks and you can only have one red block in any tower" is ambiguous, even if the agent knows the colours of the blocks (Color figure online)

(2) a. $put(o_1, o_2)$
b. "No, put green blocks on orange blocks"
c. $put(o_3, o_4)$
d. "No, put red blocks on blue blocks"
e. $put(o_5, o_6)$
f. "No, that is wrong for the same reason"

The connection *elaboration*(d,f) entails that in whatever way (2d) is disambiguated—i.e, $r_1^{r,b}$, or $r_2^{r,b}$—"the same reason" in (2f) refers to this rule. So actions (2c) and (2e) both violate the same rule, and so impose joint constraints on the colours of the four blocks o_3, o_4, o_5 and o_6. This differs from the interpretation of a similar dialogue where the agent says (3d) below:

(3) a. $put(o_3, o_4)$
b. "No, put red blocks on blue blocks"
c. $put(o_5, o_6)$
d. "No, put red blocks on blue blocks"

Utterance (3d) doesn't feature a presupposition requiring an antecedent, and so coherence does *not* demand that it be related to (3b). Thus the ambiguities in (3b) and (3d) may resolve in different ways—whether they do or not depends on the agent's current beliefs about the colours of the blocks $o_3 - o_6$.

The utterance u_5 = "that is not red either" requires an antecedent individual that's not red. With this in mind, consider dialogue (4):

(4) a. $put(o_1, o_2)$
b. "No, put orange blocks on red blocks" (teacher points at tower)
c. $put(o_3, o_4)$
d. "No, put red blocks on blue blocks" (teacher points at tower)
e. $put(o_5, o_6)$
f. "No, put purple blocks on pink blocks" (teacher points at tower)
g. $put(o_7, o_8)$
h. "No, that is not red either"

The utterance (4h) corrects action (4g), and coherence demands that it also attach to a prior utterance that entails that something isn't red. It cannot attach to (4e) with elaboration or with any other relation: in particular, it cannot elaborate either of the rules that (4e) might express while at the same time violating a rule that's expressed in terms of red, which it must do given that (4h) corrects an action (namely, (4g)). On the other hand, if the agent's beliefs about the colours of o_3 and o_4 are consistent with resolving the ambiguity in (4d) to $r_2^{r,b}$, then by (7) this interpretation provides an antecedent that's not red—namely o_3—and moreover it supports an elaboration relation between (4d) and (4h). Thus discourse coherence results in (4h) attaching to (4d) with *elaboration*, the ambiguity in (4d) is resolved to $r_2^{r,b}$, and hence (via Eq. (7)) o_3 and o_7 are not red and o_4 and o_8 are blue.

Both these examples illustrate how anaphora can impose additional constraints on interpreting both linguistic and non-linguistic moves. Our model (Sect. 5) and experiments (experiment 3 in Sect. 6) show that exploiting anaphora can help the agent to learn faster.

4.4 Silence is assent

The teacher's dialogue strategy, the knowledge of which both the teacher and learner share, assumes that silence is assent. In other words, when the teacher *doesn't* correct the agent's latest action a, then a is a part of a valid plan for achieving G without the need to undo its effects. This assumption allows the agent to use the above equations to infer the colour of unknown blocks and also which rules are *not* in the goal. Suppose, for instance, that the learner is now aware of the colour terms blue and red. So it knows that $r_1^{r,b}$ is a candidate rule in G. If its latest action a doesn't get corrected by the teacher, then either this rule is not in the goal G, or the rule is in G but was not violated by a (i.e., a red block was put on a blue one, or a non-red block was put on something). We'll present a model that exploits this in Sect. 5 and demonstrate its effectiveness in learning the task in Sect. 6 (see experiment 2).

5 System description

To solve the task described in Sect. 3, the agent we build switches between two sets of behaviour: Action Selection, where the agent uses its accumulated knowledge to construct a plan for completing the task at hand; and Correction Handling, where the agent updates its beliefs about the planning problem, to improve its ability to plan.

Algorithm 1 describes how these two systems are called upon during the solving of a single planning problem instance, while Fig. 5 describes the main components of the agent's systems and how they interact with the teacher and the world.

5.1 Action selection

To generate a valid plan, the agent uses the MetricFF symbolic planner [31, 32]). As input, the planner requires a representation of the current world state, the goal, and a set of action descriptions. The agent's initial representation of the planning problem is shown in Figs. 6 and 7. As we described earlier, these representations are deficient in that parts of the goal are missing, the predicate symbols in which the missing parts of the goal are expressed (namely, predicate symbols corresponding to colour), are not a part of the agent's conceptualisation of the domain, and the agent therefore doesn't know how to evaluate the truth value of literals involving those symbols. The action descriptions `put(x,y)` and `unstack(x,y)` are known. The remaining aspects of the planning problem (that is, the state and the goal as defined in Fig. 7) get revised as learning proceeds: predicate symbols like `red` and `blue` get added to the vocabulary of fluents (see Fig. 8), and accordingly literals such as `(red b3)` and `(blue b4)` get added to (or removed from) the state. Further, statements such as the one given in (11) (the PDDL equivalent of $r_2^{r,b}$) get added to (or removed from) the goal (see also Fig. 9).

```
forall(?y)(or(not(blue?y))(exists(?x)(and(red?x)(on?x?y))))
```
(11)

The question then is: exactly when and how do such literals and formulae become a part of the agent's representation of the planning problem, which the MetricFF must then solve? The remainder of this section describes how the Action Selection system copes with uncertainty about its (symbolic) representation of the current state and the goal G in its search for a valid plan.

5.1.1 Grounding models

The agent constructs a representation of the current state using its (current) grounding model to predict which blocks can be described by which colours. Grounding amounts to estimating how likely it is that a particular colour term can be used to describe a particular object, given its observable visual features, e.g. $P(Red(x)|F(x))$ where $Red(x)$ is a binary variable indicating whether x is red, or not. We use binary variables over a categorical *colour* distribution because more than one concept might apply to the object, such as *red* and *maroon*, and because the agent doesn't know the set of possible colours. Binary classifiers allow the seamless addition of new concepts as and when they are discovered. This probability can be estimated using Bayes Rule:

$$P(Red(x)|F(x)) = \frac{1}{\eta} P(F(x)|Red(x))P(Red(x)) \tag{12}$$

where

$$\eta = \sum_{i \in \{0,1\}} P(F(x)|Red(x) = i)P(Red(x) = i) \tag{13}$$

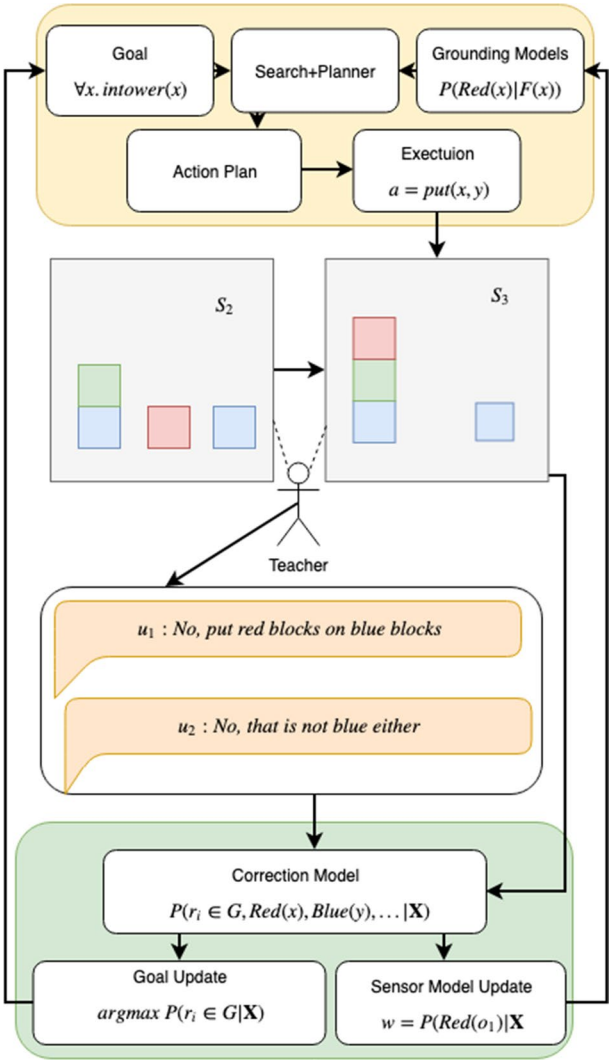


Fig. 5 The agent consists of an action selection system (yellow) and a learning system (green). Action selection uses a symbolic planner to find a plan given the most likely goal and grounding of colour terms. The learning system uses coherence to build a probability model, used to learn what rules are in the goal and how to ground colour terms

We set the prior $P(Red(x)) = 0.5$ Note that this is a binary classification between $Red(x)$ and $\neg Red(x)$ rather than a selection of a colour among several options. A more reasonable value for the prior would be $1/(\#colours)$; however, the number of colours is unknown prior to execution and so it's not possible to use this. Instead, we use 0.5 to allow the importance of the $P(F(x)|Red(x))$ term to take precedence over the prior.

We define the likelihood $P(F(x)|red(x) = 1)$ vs. $P(F(x)|red(x) = 0)$ differently. We represent $P(F(x)|red(x) = 0)$ as a uniform distribution: since there isn't a consistent signature for colours that aren't red, we expect them to be distributed fairly uniformly over

```
(define (domain blocksworld)
  (:requirements :strips
   :negative-preconditions
   :conditional-effects)
  (:predicates
    (clear ?x)    (on-table ?x)
    (arm-empty)   (holding ?x)
    (on ?x ?y)    (in-tower ?x ?t)
    (tower ?x))

  (:functions)

  (:action put
    :parameters (?ob ?underob ?tower)
    :precondition (and (clear ?ob) (on-table ?ob) (arm-empty)
                       (clear ?underob) (in-tower ?underob ?tower))
    :effect (and
      (on ?ob ?underob)      (not (clear ?underob))
      (not (on-table ?ob))   (in-tower ?ob ?tower)))

 (:action unstack
  :parameters (?ob ?underob ?tower)
  :precondition (and (clear ?ob) (on ?ob ?underob)
                     (arm-empty)(in-tower ?ob ?tower))
  :effect (and
    (on-table ?ob)
    (not (on ?ob ?underob))
    (not (in-tower ?ob ?tower))
    (clear ?underob))))
```

Fig. 6 The basic domain definition includes the actions put and unstack

```
(define (problem blocks-problem)
    (:domain blocksworld-unstack)

    (:objects b0 b1 b2 t0 t1 tower0 tower1)

    (:init
        (arm-empty )
        (on-table b0) (clear b0)
        (on-table b1) (clear b1)
        (on-table b2) (clear b2)
        (in-tower t0 tower0) (clear t0)
        (in-tower t1 tower1) (clear t1)
        (tower tower0) (tower tower1))

(:goal
    (and (forall (?x) (exists (?t)  (in-tower ?x ?t))))))
```

Fig. 7 The problem definition that the agent can access prior to interaction with the teacher and learning. It includes an initial state with the position of all objects and a goal definition which requires every block to be in a tower. This shows an example shows a problem with 3 blocks and 2 towers, but our problems use 10 blocks and 1–3 towers

Algorithm 1: The general procedure for completing a plan in each planning problem instance. The function FINDPLAN is described in Sect. 5.1, HANDLE CORRECTION in Sect. 5.2, and HANDLEASSENT in Sect. 4.4. ELICITFEEDBACK simply gives the teacher and opportunity to give feedback, either returning a correction or declining to give feedback

```
function DoScenario(state, teacher, agent, use_assent)
    while IsCompleted(state) is False do
        plan = FindPlan(state, agent)
        for a, (x, y) ∈ plan do
            state = ExecuteAction(state, action)
            u, z = ElicitFeedback(teacher, state)
            if IsCorrection(u) then
                HandleCorrection(u, a, (x, y, z))
                break
            else if IsAssent(u) & use_assent = True then
                HandleAssent(a, (x, y))
            end if
        end for
    end while
end function
```

```
function UpdateGroundingModels(P, X)
    for c(o) ∈ P do
        w = P(c(o)|X)
        if w > τ then
            UpdateParameters(c, F(o), w)
        end if
    end for
end function
```

Algorithm 2: To update the grounding models the agent takes model P and current evidence X, for every colour concept node (i.e. node which represents a particular object being a particular colour, such as $Red(o_4)$) in the model, infer the probability of that node given the available evidence. If it is above a threshold τ which we set to 0.7 then we update the parameters of the colour model for the colour c given the features of o weighted by its probability w. In our implementation UPDATEPARAMTERS adds the datapoint to the wKDE and the grounding probability is calculated using Equation (14)

the RGB spectrum. We do not update this distribution during training.[5] $P(F(x)|Red(x) = 1)$ is estimated with weighted Kernel Density Estimation (wKDE). KDE is a non-parametric model which places a kernel around every known data point and calculates the probability of a new data point by summing over the values of the kernels at that point. For wKDE this sum is weighted by weights w and normalised by their sum. For m data points $\{(w_1, F(x_1)), \ldots (w_m, F(x_m))\}$ and kernel φ this becomes:

$$P(F(x)|Red(x) = 1) = \frac{1}{\sum_{i=1}^{m} w_i} \sum_{i=1}^{m} w_i \cdot \varphi(F(x) - F(x_i)) \tag{14}$$

We use a diagonal Gaussian kernel. The pairs $(w, F(x))$ are generated by the Correction Handling system (see Sect. 5.2).

5.1.2 The goal

The agent requires a formal representation of the goal G to perform planning. The agent begins with the (correct) knowledge that it must place all blocks in a specified number of towers:

[5] We could easily update it if it made sense to do so, since the corrections will generate negative exemplars of concepts (see the negated literals in Eqs. 6–9). In fact, this could be a strength of using corrective dialogue moves, compared with *descriptions* that generally do not generate such exemplars except when the linguistic form of the utterance features negation [48].

```
function UpdateGoal(P, X)
    for r ∈ P do
        if P(r ∈ G|X) > 0.5 then
            G = G ∪ r
        else
            G = G − r
        end if
    end for
end function
```

Algorithm 3: To update the goal the agent takes the current model P and the available evidence X and predicts how likely each known rule is to be in the goal given the evidence. If the probability is higher than 0.5 then the rule is added to the goal, if it is lower it is not added and is removed if it was in the goal already

$$\forall x.(block(x) \rightarrow \exists y(tower(y) \wedge in(x, y))) \wedge count(tower) = n \tag{15}$$

However, it must use the teacher's feedback $\mathbf{X}$ to find the most likely set of additional rules which are also conjuncts in G:

$$G = \arg \max_{r_1,\ldots,r_n} P(r_1 \in G, \ldots, r_n \in G|\mathbf{X}) \tag{16}$$

The set $\mathcal{R} = \{r_1 \ldots r_n\}$ is the set of possible rules that the agent is currently aware of, as determined by the colour terms it's aware of (so $\mathcal{R}$ gets larger during learning). For each $r \in \mathcal{R}$, the agent keeps track of a probabilistic belief that $r \in G$, and assumes their mutual independence:

$$P(r \in G, r' \in G|\mathbf{X}) = P(r \in G|\mathbf{X})P(r' \in G|\mathbf{X}) \tag{17}$$

Further, the agent assumes for each $r \in \mathcal{R}$ that the prior $r \in G$ is unlikely—that is, $P(r \in G) = 0.1$. Due to the independence assumption (17), this means the goal G is constructed by adding as a conjunct any rule $r \in \mathcal{R}$ such that $P(r \in G|\mathbf{X}) > 0.5$. All other rules in $\mathcal{R}$ are omitted from G.

5.1.3 Finding a plan

The agent constructs a noisy estimate of the current state S^* and goal G given the evidence $\mathbf{X}$ so far. That noise might make it impossible for the planner to find a valid plan: for instance, it could be that according to the agent's estimates, $r_1^{r,b} \in G$ and there are more red blocks than blue blocks in S^*. In such cases, the agent recovers by searching in the probabilistic neighbourhood of S^* for alternatives from which a valid plan for achieving G can be constructed, as defined in Algorithm 4.

Algorithm 4 uses the most likely goal G throughout. It constructs a priority queue $\mathbf{S}$ of states initialised with the most likely state S^* at the top. The search proceeds by selecting the top state, S' from $\mathbf{S}$ and attempting to find a plan, returning it if successful. Otherwise the algorithm generates additional candidate search states. This is done by flipping the truth value of colour judgements, such as $Red(o_1) = 1$ to $Red(o_1) = 0$. The selection of values to flip are made to maximise the probability of the state and to move towards states which are more likely to allow for a valid plan, for example, if $r_1^{r,b} \in G^*$ then values are selected to increase the number of blue blocks or decrease the number of red blocks. The algorithm continues until a plan is found, or a maximum number of 'flipping' steps N have been taken, in which case the agent constructs a plan that satisfies the default goal of

```
function FindPlan(state, agent)
    S* ← {C, o : C(o) = argmax_{i∈{0,1}} P(C(o) = i|F(o))}
    S ← priorityQ((P(S*), S*))
    G ← arg max_{r_1,...,r_n} P(r_i ∈ G| X )
    for i ← {1 : N} do
        S' ← S.pop()
        plan ← Planner.find_plan(S', G)
        if plan ≠ None then
            return plan
        else
            S.push(generate_search_states(S', G*))
        end if
    end for
return Planner.find_plan(S^0, G^0)
end function
```

Algorithm 4: Algorithm for finding a plan that is most likely to be valid, given thecolours the agent is currently aware of and its beliefs about symbol grounding (ie, beliefs about which blocks are which colours) and its beliefs about the goal. For the planner we use MetricFF. *N* represents the number of states that the learner searches through before resorting to the default plan, of simply putting all blocks in the specified number of towers, regardless of their colour. *N* is a hyper parameter, which we have set to 20 for our experiments. S^* is the state made up of the most likely decision about the colour of each block for all known colours and objects

having all blocks in towers, arranged in any way regardless of their colour. This ensures some plan is always found.

Once a plan is found the agent executes its actions in sequence until either the problem instance is completed, or the teacher gives corrective feedback. If a correction occurs then the Correction Handling system takes over.

5.2 Handling corrections

As shown in Sect. 5.1, the two main outcomes expected from correction handling are updating the grounding models and the goal *G*. We do this by making use of the constraints imposed by equations 5–9 in Sect. 4.

5.2.1 Adding to the domain model

When the teacher provides feedback, her utterance u is parsed using the English Resource Grammar [14] to extract relevant content words. The surface form of the utterance together with the extracted content words are used to construct rules in the shape of r_1, r_2, and r_3 (the mapping between input utterance and rules is hand crafted, and shown in Table 1). The teacher's pointing action (a symbolic specification of the block being pointed at, either a block in the tower directly violating a rule or a block on the table) tells the agent whether a direct or indirect violation has occurred. When the teacher's feedback features an anaphoric expression (see Sect. 4.3), this stage also looks back in the dialogue to find which previous utterance it should connect to. The output of this step reveals which possible messages $M(u)$ the speaker could have intended, given the syntactic form of her utterance—the three possible choices are (1) r_1 or r_2 (e.g., "no, put red blocks on blue blocks"); or (2) r_3 (e.g., "no, you can have at most one red block in any tower"); or (3) r_3 combined with either r_1 or r_2 (e.g., "no, you can have at most

Fig. 8 Updating the domain by adding the new colour concepts the agent has become aware of

```
(define (domain blocksworld)
    (...)
    (:predicates (...)
    (blue ?x) (red ?x))

    (...))
```

```
(define (problem blocks-problem)
(:domain blocksworld-unstack)
(...)
(:init
(...)
(blue b0) (red b1))

(:goal (and (...)
    (forall (?y) (or (not (blue ?y))
    (exists (?x) (and (red ?x) (on ?x ?y))))))))
```

Fig. 9 What the agent adds to the problem initial state if the agent is told "no, red blocks should be on blue blocks and it infers that $r_2^{r,b}$ is in the goal. Adding (*blueb*0) and (*redb*1) to the initial state assumes the agent's grounding models have deemed these objects to have a probability of being each of those colours bigger than 0.5 and all other objects not to satisfy this probability threshold (Color figure online)

one block in any tower and put red blocks on blue blocks"), where in each case the relevant colour and number terms that feature in the speaker's utterance (correctly) annotate those possibilities. It also reveals whether the violation is direct or indirect (via the pointing), and whether the utterance featured an anaphoric expression.

At this stage, the agent will check if it has encountered before the colour words which are referenced in the correction or not. If a word "C" has not been encountered before, then the agent adds C to its NL vocabulary, the binary random variable $C(x)$ to its domain model, and generates a new grounding model, for estimating $P(C(x)|F(x))$ for the previously unforeseen concept C. It also extends its set of possible rules $\mathcal{R}$ to include those rules that feature C. Figures 8 and 9 shows how the domain and problem files are updated when the agent encounters the concepts *red* and *blue* and infers $r_2^{r,b}$ is in the goal.

5.2.2 Inferring the most likely state

The agent updates its grounding model by inferring which of the states in Figs. 2, 3 and 4 is most likely, and therefore, which blocks have which colour. Making this inference also allows the agent to infer which rules are in the goal. In this section we consider how the agent makes these inferences after an example correction u_1 = "no, put red blocks on blue blocks" of the action $a_1 = put(o_1, o_2)$. When appropriate we will differentiate between a direct violation (indicated by the teacher pointing at the tower) as u_1^D and an indirect violation (indicated by the teacher pointing at another block o_3) as u_1^I. The evidence which the agent observes for a correction is twofold. First, the agent knows a correction occurred: i.e. $Corr(a_1, u_1) = True$. Secondly, the agent can observe $F(o_1)$, $F(o_2)$, and $F(o_3)$.

Given the available evidence the agent attempts to make two types of inference given the available evidence: (1) which of the rules is most likely to be in the goal:

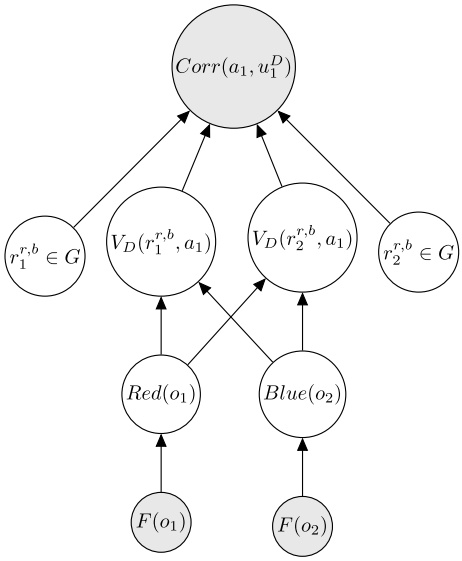


Fig. 10 The nodes added to the probabilistic graphical model after a correction $u_1^D =$ "no, put red blocks on blue blocks". Grey nodes are observed and white ones are latent (Color figure online)

$$P(r_i \in G | Corr(a_1, u_1), F(o_1), F(o_2), F(o_3)) \tag{18}$$

and (2) which block can be described with a relevant colour

$$P(Red(o_1) | Corr(a_1, u_1), F(o_1), F(o_2), F(o_3)) \tag{19}$$

$$P(Blue(o_2) | Corr(a_1, u_1), F(o_1), F(o_2), F(o_3)) \tag{20}$$

For any particular correction the probability of a rule being in the goal will only change for those rules in the set $M(u)$, due to our independence assumption. Further, the agent can only make inferences about the colour of blocks which have been involved in the correction, i.e. $Red(o_1)$, $Blue(o_2)$, and, if it is an indirect violation, $Red(o_3)$ and $Blue(o_3)$.

To compute these inferences, the agent builds a probabilistic model which captures the semantics of Eqs. 5–9. The model generated for a direct violation is pictured graphically in Fig. 10: grey nodes represent observed evidence while white nodes represent latent variables. The arrows represent probabilistic dependencies. Starting at the top, the factor (21) is defines so as to capture Eq. (5):

$$P(Corr(a_1, u_1) | r_1^{r,b} \in G, V_D(r_1^{r,b}, a_1), V_D(r_2^{r,b}, a_1), r_2^{r,b} \in G) \tag{21}$$

It is a deterministic factor, which provides a probability of 1 to any state which satisfies Eq. (5): so when $Corr(a_1, u_1) = True$ one of the two rules—$r_1^{r,b}$ or $r_2^{r,b}$—must be violated and part of the goal G. Any other state is given 0 probability. Thus an interpretation of the evidence where neither rule is violated or neither rule is in G is ruled out.

Next are the violation factors $V_D(r_1^{r,b}, a_1)$ and $V_D(r_2^{r,b}, a_1)$. The relevant factor (22) is determined deterministically via Eq. (6) for $i = 1$ and (7) for $i = 2$:

$$P(V_D(r_i^{r,b}, a_1) | Red(o_1), Blue(o_2)) \tag{22}$$

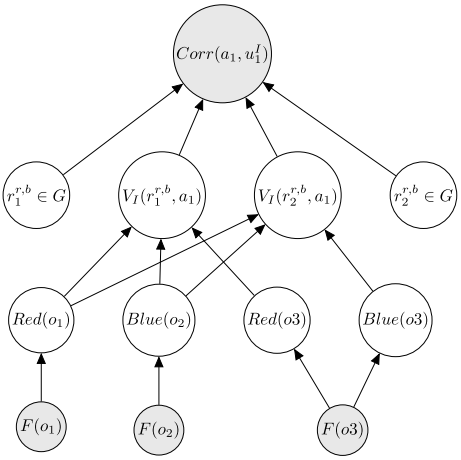


Fig. 11 The nodes added to the graphical model after an indirect violation u_1^I = "no, put red blocks on blue blocks" with the teacher pointing at o_3. Grey nodes are observed and white nodes are latent (Color figure online)

In other words, $P(V_D(r_1^{r,b}, a_1) = True | red(o_1), \neg blue(o_2)) = 1$, and for any other values of $Red(o_1)$ and $Blue(o_2)$ the probability is 0. For $V_D(r_1^{r,b}, a_1) = False$ the exact opposite is the case. For $V_D(r_2^{r,b}, a_1)$ the same holds but for Eq. (7) instead.

The remaining nodes $P(Red(o_1)|F(o_1))$ and $P(Blue(o_2)|F(o_2))$ are defined by the agent's current grounding model. $P(F(o_i))$ is a prior which is assumed to be a constant for all o_i. Finally, $P(r_i^{r,b} \in G)$ is the agent's prior belief that $r_i^{r,b}$ is in the goal ($i = 1, 2$). As we mentioned earlier, this is set to 0.1 initially; however, the prior will have been updated by what was learned from earlier problem instances where the agent was trying to learn this planning problem. The details of this are specified in Sect. 5.2.7.

When the teacher's pointing indicates an *indirect* violation, resulting in utterance u_1^I, the agent's aim is the same: to infer the most likely rule that the teacher is expressing (i.e., $r_1^{r,b}$ or $r_2^{r,b}$) as a part of the goal G, and to update its inferences about the colour of relevant objects in the current state. Equations 8 and 9 constrain the colour of the top two blocks in the tower and relative number of those remaining on the table. We could generate a probabilistic model which estimates the probability of each block being red and blue, which could provide several new data points for the grounding models. However, doing so carries a considerable computational cost. In the worst case, for a problem instance with 10 blocks in it, there might be 8 blocks left on the table. The model would then require 16 binary variables to represent each block possibly being either red or blue. This would mean 2^{16} possible interpretations of the world which would be infeasible to deal with. As such we have chosen to simplify the inference problem while still supporting efficient learning. The simplification is that, out of the blocks on the table, we will only consider the colour of the block which the teacher pointed at, in our example o_3. We do this by taking into account the fact that the object the teacher is pointing at will be red for $V_I(r_1^{r,b}, a_1)$ and blue for $V_I(r_2^{r,b}, a_1)$. This yields new equations for interpreting Indirect violations:

$$V_I(r_1^{r,b}, a_1) \leftrightarrow (\neg red(o_1) \wedge blue(o_2) \wedge on(o_1, o_2) \wedge red(o_3)) \tag{23}$$

$$V_I(r_2^{r,b}, a_1) \leftrightarrow (red(o_1) \wedge \neg blue(o_2) \wedge on(o_1, o_2) \wedge blue(o_3)) \tag{24}$$

Having established these equations, building a model capturing a semantics is straight forward, as shown in Fig. 11. Just as with the direct violation model, $P(corr(a_1, u_1))$ is

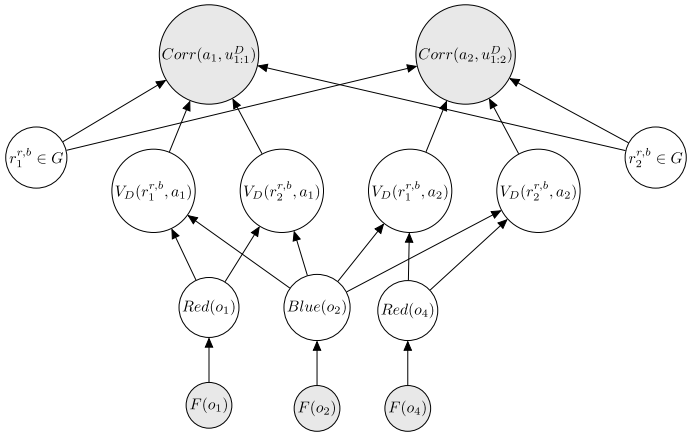


Fig. 12 The state of the graphical model after two corrections: first $a_1 = put(o_1, o_2)$ corrected with $u^D_{1:1}$ = "no, put red blocks on blue blocks" followed by $a_2 = put(o_4, o_2)$ being corrected by $u^D_{1:2}$ = "no, put red blocks on blue blocks" (Color figure online)

assigned a probability of 1 only for cases where $r_1^{r,b} \in G$ and $V_I(r_1^{r,b}, a_1)$ is true, or $r_2 \in G$ and $V_I(r_2^{r,b}, a_1)$ is true. The violation variables then gives a probability of 1 to any interpretation that fits with Eqs. 23 and 24 respectively; 0 otherwise. The remaining parameters of Fig. 11 are the same as described above.

5.2.3 Repeated corrections and anaphora

The agent may be wrong more than once in the course of solving its planning problem, leading to several occasions where the teacher utters a correction. In Appelgren and Lascarides [5] we treated each such correction as a separate event, disconnected from prior corrections except through the probabilistic updates to the grounding model and to the goal G. This meant that the agent would not revise any *previous* inferences about the teacher's intended message, given subsequent evidence. In our new model, we fix this by cultivating a growing probabilistic model. Each time a new correction is given, additional nodes are added to the existing model, reflecting the new evidence and new latent information. For example, if after a_1 and u_1, the top object o_1 is removed and the agent performs the action $a_2 = put(o_4, o_2)$ and the teacher utters the same correction, i.e. u_2 = "no, put red blocks on blue blocks", then nodes for $F(o_4)$ and $Red(o_4)$ will be added, together with violation variables and a correction variable, resulting in Fig. 12. However, importantly, $Blue(o_2)$ and $F(o_2)$ as well as $r_1^{r,b} \in G$ and $r_2^{r,b} \in G$ will connect to the newly added correction variables. Inference is done on the entire model, which may grow large depending on the number of mistakes the agent makes. However, given the incremental nature of the feedback it is possible to keep the inference reasonably small by keeping track of which possible latent variable states had non-zero probability previously. This set will be relatively small: for example, for the correction u_1^D there are only 4 non-zero probability latent states, compared to $2^6 = 64$ possible latent states. Combined with the fact that no previously discounted (0 probability) state can ever be made non-zero, it means that when a second correction is

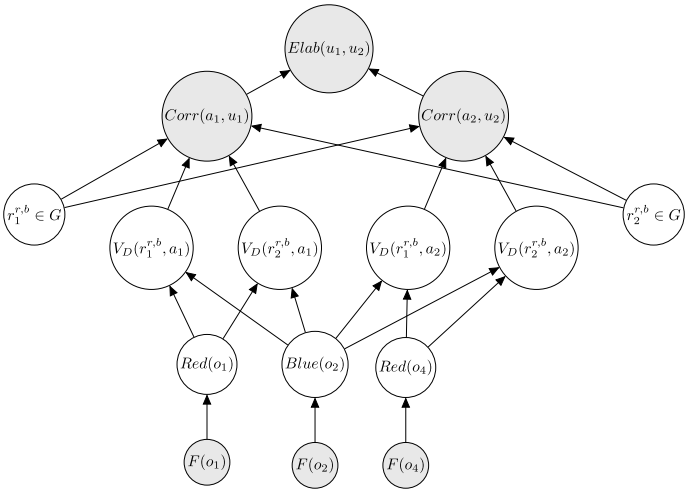


Fig. 13 The graph after two corrections where the second is an elaboration

added, only these 4 latent states need to be considered, together with any states added by the new correction.

In this work we also introduce the possibility for the teacher to use additional corrections which contain anaphoric expressions, instead of repeating the same correction content several times. These elaborations of prior corrections, introduced in Sect. 4.3, will also be used to update our probability model. As established in that section, an elaboration implies that the content of the previous correction also applies to the current action. This means the first step to dealing with the elaborations is adding to the model exactly the nodes which would have been added if the teacher had instead repeated the correction. However, the additional information which the agent can exploit during learning is that the violated rule is the same in both situations. By default, the model allows an interpretation of two corrections that are not coherently related to violate distinct rules, *even if* the utterances are the same—"no, put red blocks on blue blocks", say. It must do this, because both $r_1^{r,b}$ and $r_2^{r,b}$ may be in G and the teacher uses the same (ambiguous) utterance to describe both rules. So, to ensure that the agent's inferences are consistent with the fact that two corrections are connected with the coherence relation *elaboration* (or *Elab* for short), we add a Boolean random variable $Elab(u_1, u_2)$, where (25) has a non-zero probability only if both $V(r_1^{r,b}, a_1) = True$ and $V(r_1^{r,b}, a_2) = True$, or $V(r_2^{r,b}, a_1) = True$ and $V(r_2^{r,b}, a_2) = True$.

$$P(Elab(u_1, u_2) | V(r_1^{r,b}, a_1), V(r_2^{r,b}, a_1), V(r_1^{r,b}, a_2), V(r_2^{r,b}, a_2)) \tag{25}$$

The resulting probability model is shown in Fig. 13.

For "no, that is not red either" the same elaboration variable is added to the graphical model. Additionally, the agent will know that an object was "not red". It can infer which object it is by reasoning that the redness of that object must have bearing on the inference. In other words, o_2 not being red is not relevant to the correction u_1, while knowing that o_1 or o_4 is is highly valuable. From this fact the agent can infer that these are the objects that must have been "not red" and it therefore adds $red(o_1) = 0$ and $red(o_4) = 0$ to its observed evidence, which allows the agent to infer the correct latent states for two different actions.

function HANDLEASSENT(a, (x, y))
  $X_t = X_{t-1}$
  for $r \in R$ **do**
    $c_1, c_2 = r.colours$
    if $P(c_1(x)|F(x)) > \tau \vee P_t(c_2(y)|F(x)) > \tau$ **then**
      $P_t =$ BUILDMODEL(P_t, r, (x, y))
      $X_t = X_t \cup \{Corr(u, a) = False, F(x), F(y)\}$
    end if
    UPDATEGROUNDINGMODELS(P_t, X_t)
    UPDATEGOAL(P_t, X_t)
  end for
end function

Algorithm 5 The algorithm for handling a corrections. The input to the function are the teacher's utterance μ, the most recent action a, and the arguments to this action x and y, if the teacher pointed at a block on the table the id of that block is provided as z. Also included in the function are two persistent structures: P_t representing the probability model and X_t representing the known observable evidence at time t. The agent extracts the possible messages $M(\mu)$ from the teacher's utterance μ. It keeps track of the colours it knows about in C and the rules it knows about in R which can easily be extracted from the messages. It uses the message and the arguments of the action to update the probability model. X_t contains the observed data. The agent checks if it can infer which rule was violated with probability higher than τ which we set as 0.7. If not the agent asks for help, and the answer is added to the available evidence. Lastly the agent updates its grounding models and goal description and undoes the offending action

5.2.4 Inferences when no correction occurs

As described in Sect. 4.4, the shared dialogue strategy between teacher and agent means that the teacher's silence implies assent—the agent's latest action a is part of a valid plan for achieving G. We can take advantage of this fact, by using Eqs. 6 and 7: we know that neither of these rules has been violated. Therefore, when no correction is given the agent could add the same nodes as it would for when a correction happens, i.e. those in Fig. 10. Then, instead of the observed evidence $Corr(u_1, a_1) = True$, the observed evidence is $Corr(u, a_1) = False$ (where u = silence). The agent can then perform updates to its model of symbol grounding, as before.

Doing this for every uncorrected move and the entire set of known rules would become very computationally expensive as the model would grow very large. Further, the benefit will be minor in cases where no correction is *expected*, i.e., when blocks of colours not relevant to known rules are placed on each other. Therefore, the biggest benefit will be gained when a block with a colour relevant to a rule is involved in the action. For example, if a red block is placed, the agents estimates for rules such as $r_1^{r,x}$, $r_2 r, x$ and $r_3^{r,n}$ could be changed. If, on the other hand, a non-red block is placed, the agents belief about these rules would not be impacted. As such, we keep track of which blocks the agent has a strong belief about their colour and only perform an update when these blocks are part of an action, making inferences using the relevant violation factors for impacted rules. We use a threshold $P(Colour(o_1)|F(o_1)) > 0.7$ to trigger this behaviour.

5.2.5 Colour count rules

Dealing with rules of type r_3 requires a way to update the model in the two situations described in Sect. 4.2. In the case where, for example, $r_3^{r,1} \in G$ and the agent places a

```
(define (domain blocksworld-unstack)
    (...)
    (:predicates
        (...) (blue ?x))

    (:functions (blue-count ?tower))

    (:action put
        (...)
        :effect (and (...)
            (when (blue ?ob) (increase (blue-count ?tower) 1))))

    (:action unstack
        (...)
        :effect (and (...)
            (when (blue ?ob) (decrease (blue-count ?tower) 1)))))
```

Fig. 14 When the agent learns that $r_3^{blue,3}$ constrains the problem, the agent adds the colour concept `blue` to the domain, it additionally adds `(blue-count ?tower)` as a function. This will be used to count the number of blue blocks in a tower. We also update the actions to keep track of these counts (Color figure online)

```
(define (problem blocks-problem)
    (:domain blocksworld-unstack)
    (...)
    (:init
        (...) (blue b0) (blue b5)
        (= (blue-count t0) 0)
(= (blue-count t1) 0))
    (:goal (and (...) (forall (?t) (or (not (tower ?t))
        (<= (blue-count ?t) 1))))))
```

Fig. 15 When the agent learns that $r_3^{blue,1}$ constrains the problem the agent adds `(= (blue-count tx) 0)` for every tower in the problem as well as adding the actual r_3 rule to the goal description

second red block on a tower, the teacher's correction will result in the agent constructing the graphical model in Fig. 16. It has a similar structure to the previous models, except that it includes a node for every block in the relevant tower: this enables the model to capture inferences that at least one block in the tower already in the tower must be red (see Eq. (10)). The factor $P(Corr(a,u)|r_3, V(r_3,a))$ is defined in a similar way to previous models, except in this case, thanks to the lack of linguistic ambiguity, there is only one possible rule. As such the probability is 1 only when the random variables $r_3^{r,1} \in G$ and $V(r_3^{r,1}, a)$ are both true; and 0 otherwise.

For the violation factor $P(V(r_3,a)|\mathbf{X})$ where $\mathbf{X}$ is the relevant evidence, this factor gives probability of 1 to any interpretation where there are exactly $n+1$ blocks of the offending colour in the tower, where n is the number in the rule, with the block that a just added to the tower being one of those (see Eq. (10)). The remaining factors are the same as in previous sections. The PDDL specification of the planning problem, which the MetricFF planner must then solve, are shown in Figs. 14 and 15: they respectively add the new colour predicate symbol and count functions to the domain model, and the PDDL representation of the r_3 rule to the goal.

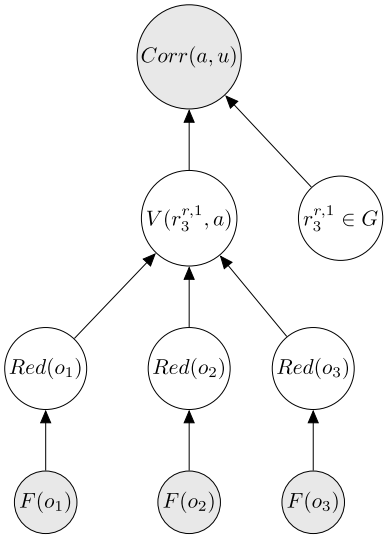


Fig. 16 The graphical model after a correction of $a_1 = put(o_1, o_2)$ of the form $u_1 =$ "no, no more than one red block in the tower". The tower at the time of correction consists of blocks o_1, o_2, and o_3 (Color figure online)

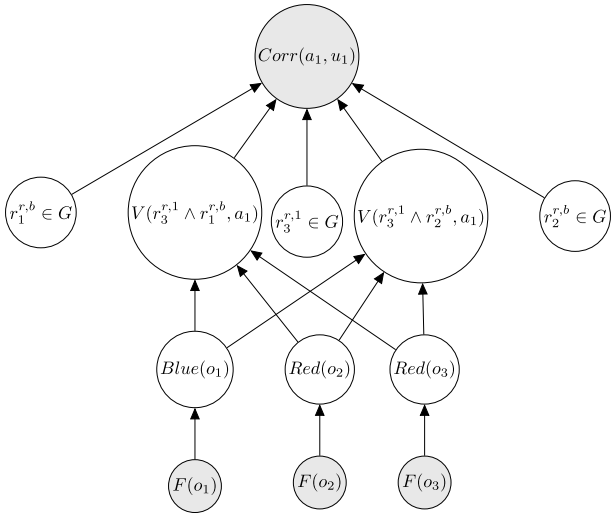


Fig. 17 The graphical model after a correction $u_1 =$ "no, no more than one red block in the tower and put red blocks on blue blocks". At the time of the correction the tower consists of o_1, o_2, and o_3 (Color figure online)

For the second violation type, we have a conjunction of two rules being violated, but neither violated on its own. Thus the graphical model that the agent generates when the teacher utters u_3 (e.g., "no, red blocks should be on blue blocks and you can have only one red block in any tower") is like Fig. 16, but with added variables $V(r_1^{r,b} \wedge r_3^{r,1}, a)$ and $V(r_2^{r,b} \wedge r_3^{r,1}, a)$; see Fig. 17. As described in Sect. 4.2, it would be possible in principle to estimate the counts of red and blue blocks on the table to inform estimates as to which of these two rules is violated, but it is computationally costly and of limited benefit. So the

factors in the model are the same regardless of whether the rule is of type $r_1^{r,b}$ or $r_2^{r,b}$: it gives probability 1 to an interpretation where the number of red blocks in the tower is equal to n and the top block is blue. Other corrections will have to be used to infer which of the two alternative rules is actually part of the goal (or both).

Figures 14 and 15 show how the domain is updated when the agent learns that $r_3^{blue,1}$ is in the goal. Since the planner will have to keep track of the number of blue blocks in a tower we need to add a function which keeps track of the this number and update the action description to update this number.

5.2.6 Using inference to update beliefs

The above graphical models, which are generated by the agent as it observes the teacher's responses to its moves, are used to update the agent's beliefs about symbol grounding and the goal G. The inference from Eq. (18) will update the agents belief about which rules are in the goal; see Sect. 5.1.2.

To update the grounding models the agent performs the inference in (26) (and similar updates for other colours):

$$w := P(Red(o_1)|\mathbf{X}) \tag{26}$$

It will then add to the wKDE models available data the new weight, data point pair $(w, F(o_1))$. In this way the model learns to better recognise objects that are "red" as it executes its plans and acquires further feedback from the teacher. However, to minimise the noise within the agent's training data, we set a threshold of for including data points: the agent updates its grounding model with its estimates of the colours of blocks in the current state only if $w > 0.7$.

In some cases the agent will not have enough previous knowledge to make a meaningful inference. For example, if the agent has not encountered the terms "red" or "blue" before there will be an equal probability placed on either of the two possible interpretations. In this case the agent will ask for help from the teacher as described in Sect. 4.1. The answer to the question will be added to the agent's observed evidence, which then triggers updates to the inferences. Due to the nature of the constraints on coherent discourse that we described in Sect. 4, this will lead to the agent being certain about which situation it is in, thereby providing a starting point for the agent's further learning of the rules and the grounding of colours.

5.2.7 New planning problem instance

Our experiments in Sect. 6 are set up so that for each goal G, the agent faces 50 planning problem instances, each constrained by the same rules, in which the agent gathers evidence for learning G and symbol grounding. Each problem instance consists of a initial world state—i.e., a fixed set of blocks on the table, the the colours of those blocks varies from one problem to another (see Sect. 6 for details). The agent attempts to complete a valid plan in each of the 50 problem instances in turn. The agent updates its beliefs about G and symbol grounding from its observations of the teacher's reaction to each action it executes, as described in this Section. Thus within any given problem, the agent's revised beliefs about G and about the world state may trigger, via the Action Selection module (Sect. 5.1), revisions to the plan it executes. When the agent encounters a new problem, the agent retains

its current beliefs, as learned from the previous ones, about symbol grounding and about G. These current beliefs are treated as *priors* when starting a new problem instance. However, the agent does *not* retain the prior problem's graphical model, which the agent generated dynamically, as the teacher-agent interaction unfolded; rather, it starts with a 'clean slate' (while retaining its beliefs about what the set of random variables are). To put this another way, we assume that the start of each problem instance marks the start of a new coherent conversation. Therefore, anaphoric expressions cannot refer back to antecedents introduced in any prior instance. This not only respects constraints on discourse coherence, but also serves to ensure the graphical model does not grow impossibly large for practical learning and inference.

6 Experiments

We present four sets of experiments, which test the methods described in this paper. The first experiment shows the value of the rich linguistic evidence we described in Sects. 3 and 4. We show that our models of language interpretation and grounding allow much faster learning for solving the planning problem than attempting to learn only from the fact that the teacher corrected the agent—in effect, learning from only the cue phrase "no". The second experiment shows that exploiting the teacher's silence as evidence of assent greatly improves the agent's speed of learning. The third experiment tests the hypothesis that exploiting anaphoric expressions of the kind we described in Sect. 4.3 when the agent repeats mistakes leads to faster learning, because coherence constraints on anaphora reduce ambiguity overall. In the final experiments we use r_3 rules to investigate the trade-off between the computational cost of exploiting monotonic entailments from the teacher's linguistic move versus the benefit to task learning that the entailment provides.

To test each of our agents, we conduct an experiment that consists of 50 planning problems or *trials*: i.e., 50 goals G. Each goal G is constructed so that there are no contradictory rules, given ground truth about the denotations of colours (e.g., $r_1^{red,blue}$ and $r_1^{red,green}$ can't both be conjuncts in G). Other than this, each of the 50 goals is randomly generated to contain a certain number of rules of certain types—we motivate the number and type of rules in our experiments according to which hypotheses we're testing, as detailed in Experiments 1 to 4 below.

For each goal G, the agent learns G and symbol grounding via 50 *planning problem instances*, which the agent tackles in sequence (see Sect. 5.2.7). For each instance, we generate an *initial state* via random sampling of 10 coloured blocks, so that (a) there is a valid plan for achieving G from that initial state; and (b) the probability distributions for choosing the colours of the 10 blocks makes it highly likely that there are several blocks with colours that are relevant to the goal G (e.g., if $r_1^{red,blue}$ is a part of G, then we make it highly likely that the initial state features red and blue blocks). At the start of each trial, a new agent is initialized—i.e., it not only has no knowledge of the goal G that it must learn, but it is also made unaware of all colour terms and their denotations. It proceeds to learn via its actions and teacher feedback, as described in Sect. 5.

We measure the performance of each agent by measuring its regret, which we define as the number of mistakes an agent makes, which is equivalent to the number of times the teacher must correct it—so low numbers are better than high ones! In the graphs below, the y axis is the average regret over the 50 goals (or trials) G that the agent faced in the experiment, and the x axis is the number of planning problem instances it has been exposed to so

far within a given trial. Significance among the different performances of two distinct agent types is tested using a paired t-test on their final total regrets over all the trials in the experiment. To try visualise this pairwise difference in terminal regret (i.e. regret after the 50 trials) we present box plots Stryjewski [58] of the difference between two agents. The box shows the median (centre line), 25th and 75th quartile (edges of the box) and the min and max value (lines on the "whiskers" coming out of the box). Individual points outside of these extremes are outliers, which are points 1.5 times away from the inter-quartile range. The difference is calculated such that positive values represent that the agent named second in the legend has higher terminal regret, i.e. performed worse. In other words, positive values are good for the agent named first and negative values are good for the second agent.

Each experiment has been run afresh for this paper with new goals and initial states; however, some experiments do overlap with previously published results. Experiment 1 mirrors the experiments presented in [5], but with an updated version of the Language agent.[6] Further, this experiment now also features goals G that draw from the additional form of rules considered in this paper, namely r_3. Experiments 2 and 3 mirror those presented in Appelgren and Lascarides [4]. Experiment 4 is completely new. Further, the manner in which colours of the blocks in the problem instances are generated has changed from previous papers. Previously, colours were selected from a fixed set of specific RGB values, whereas in this paper colours are generated from a continuous range.

6.1 Experiment 1

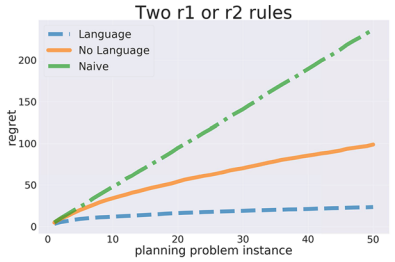
Previous work where agents learn via an interaction with a teacher made use of corrective feedback with little linguistic content [39, 51]. The purpose of our first experiment is to show that making use of linguistically more complex corrections provides valuable evidence, even when that complexity exhibits neologisms and semantic scope ambiguities that must be resolved via context in order to solve the planning problem. We show that such evidence can nevertheless enable the agent to learn faster and more effectively than a simpler teacher utterance, such as "no". So to test our hypothesis, we compare our *Language agent*, which uses the models from Sect. 5, to two baselines: the *Naive Agent* and the *No Language Agent*. The Naive Agent acts as a lower-bound on performance by not learning at all between problem instances. It simply avoids repeating any action which was previously corrected within the current problem. In other words. if $put(o_1, o_2)$ is corrected, it will not repeat that action in the same problem instance.

The No Language Agent learns from the fact that the teacher corrected the agent, but ignores its linguistic content. It observes that the teacher said "no" and whether the teacher pointed at the tower or at a particular block on the table. It makes all its inferences from these facts alone. Its Action Selection system is as described in Sect. 5.1, but its Correction Handling module is different. The No Language Agent attempts to learn rules and ground symbols. However, it will not always be able to construct full rules of form r_1 or r_2. For a direct violation the agent can only infer that objects with similar

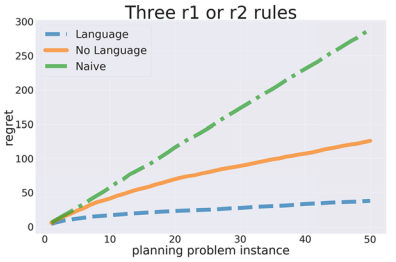
[6] Appelgren and Lascarides [5] treat each correction as an independent event while Appelgren and Lascarides [4] and this paper consider the connection between corrections by building up the model as described in Sect. 5.2.3.

Table 3 Mean Terminal Regret for for the three agents and three planning problems of Experiment 1. A pairwise t-test combining all trials shows that the No Language agent significantly outperforms the Naive Agent ($t = 37.37$, $p = 1.42e^{-77}$) And that the Language agent significantly outperforms the No Language agent ($t = 20.32$, $p = 9.39e^{-45}$)

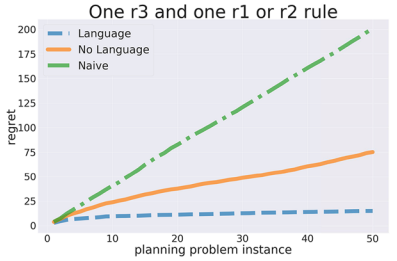
Problem	Naive	No language	Language
r_3 and r_1 or r_2	200.4	75.1	15.14
Two r_1 or r_2 rules	241.8	100.0	25.7
Three r_1 or r_2 rules	288	126	38



(a) All goals in this trial contain two r_1 or r_2 rules



(b) All goals in this trial contain three r_1 or r_2 rules



(c) All goals in this trial contain one r_3 rule and one r_1 or r_2 rule.

Fig. 18 Average cumulative regret on three different trials comparing our Language Agent, a No Language Agent, and a Naive Agent. The graphs show the mean of regret over 50 trials

RGB values to o_1 must not be placed on objects similar to o_2, as shown in (27), where C_{o_i} is grounded by creating a new grounding model with a single data point, namely $F(o_i)$:

$$\neg\exists x \cdot y \cdot C_{o_1}(x) \wedge C_{o_2}(y) \wedge on(x, y) \tag{27}$$

When the teacher points at o_3 it's an indirect violation of an r_1 or r_2 rule, so the agent knows that o_3 must either be placed on objects similar to o_2 or that objects similar to o_1 must be placed on o_3:

$$r_1 = \forall x \cdot C_{o_3}(x) \rightarrow \exists y \cdot C_{o_2}(y) \wedge on(x, y) \tag{28}$$

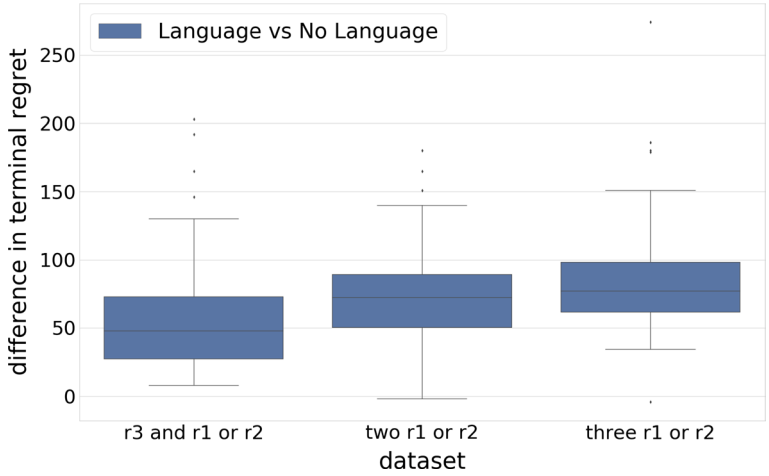


Fig. 19 This box plot shows the difference in terminal regret between the Language agent and the No Language agent. Positive values mean the No Language agent has higher terminal regret than the Language agent

Table 4 Mean Terminal Regret for agent with Correction Only (CO), and Use Assent (UA) with or without Anaphor (A)

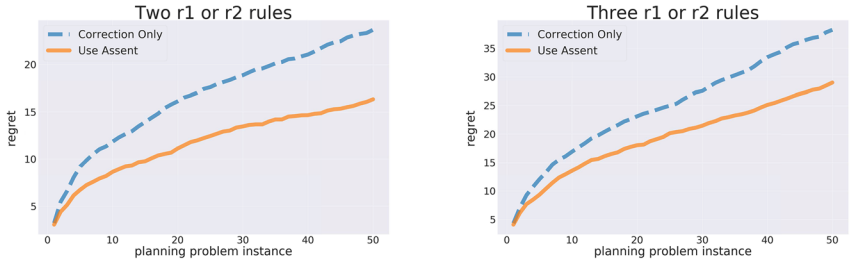
Problem	CO	UA	CO+A	UA+A
Two r_1 or r_2 rules	23.6	22.3	16.3	15.4
Three r_1 or r_2 rules	38.3	35.5	29.0	26.7
Both r_1 and r_2	17.7	12.8	11.8	10.4

A pairwise t-test shows using assent significantly improves

$$r_2 = \forall y \cdot C_{o_3}(y) \rightarrow \exists x \cdot C_{o_1}(x) \wedge on(x, y) \tag{29}$$

However, the agent does not know which is correct. It attempts to find out by attempting to break one of the rules. If r_1 is in the goal then placing a block different from o_3 on o_2 should cause the teacher to correct it. Thus the agent finds the remaining block most dissimilar to o_3 and places it on o_2. If a correction occurs then the agent adds r_1 to the goal; otherwise, r_2. Finally, when no pointing occurs, the violated rule must be of type r_3. The agent knows C_{o_1} must be the constrained colour in the rule, but it does not know the number n. However, it estimates n by counting the blocks in the tower that are similar to o_1, thereby estimating the violated rule to be $r_3^{C_{o_1},n}$.

We test these three agents in three sets of 50 trials: these three sets differ in the number of rules that are present in its 50 goals (2 or 3 rules) and in the types of those rules, as shown in the results in Fig. 18 with average terminal regret summarised in Table 3 and visualised in Fig. 19. The No Language Agent significantly outperforms the Naive Agent ($t = 37.37$, $p = 1.42e^{-77}$), showing that there is some merit to it. However, the No Language agent is, in turn, significantly outperformed by the Language Agent ($t = 20.32$, $p = 9.39e^{-45}$), showing that the evidence provided by the teacher's linguistic feedback is extremely valuable, this is also supported by the fact that the Language agent always has a lower terminal regret as shown in Fig. 19. In addition to the



(a) In this trial each goal contains two r_1 or r_2 rules.

(b) In this trial each goal contains three r_1 or r_2 rules.

Fig. 20 Experiment comparing an agent updating only on corrections against updating on the teacher's silence, which implies assent. Results are mean regret over 50 trial

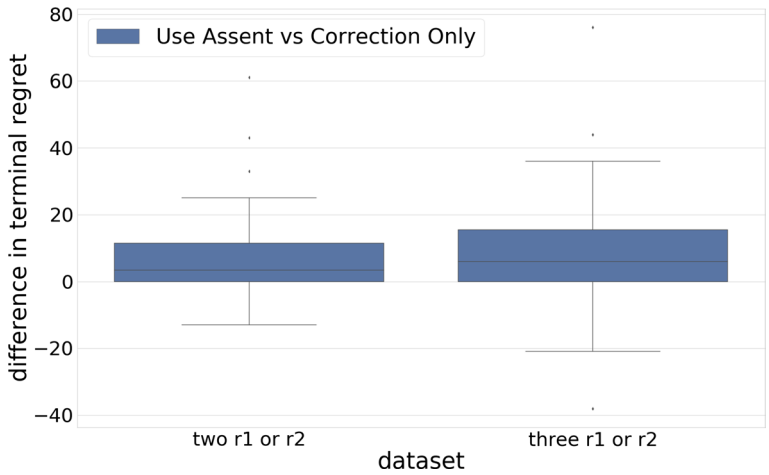
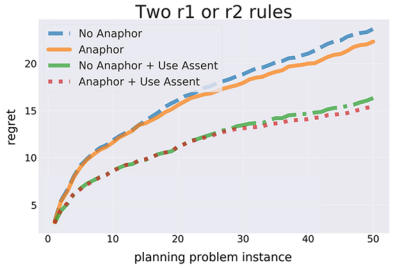


Fig. 21 This box plot shows the difference in terminal regret between using assent and not (correction only). Positive values indicate that correction only has higher terminal regret than using assent

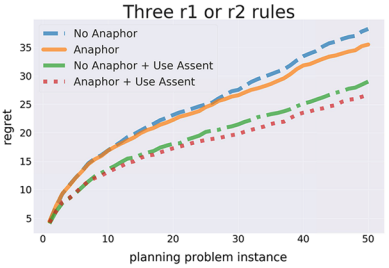
Language Agent's enhanced performance, it also learns to increase the common language with the teacher. In a broader ITL setting, this knowledge of grounding can allow transfer learning to new tasks [38].

6.2 Experiment 2

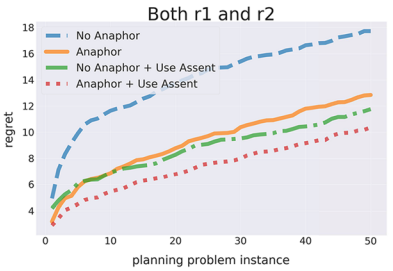
Our second experiment tests the hypothesis that exploiting the fact that the teacher is silent, thereby implying assent, enhances data efficiency. We compare the Language Agent from Experiment 1, which does not update its model when there's implied assent, to an agent that does, as described in Sect. 5.2.4. We compare the two agents in two different sets of 50 trials, one where the goals consist of two r_1 and/or r_2 rules, and one where the goals have



(a) In this trial each goal contains two r_1 or r_2 rules.



(b) In this trial each goal contains three r_1 or r_2 rules



(c) In this trial each goal contains a pair of r_1 and r_2 rules with the same pair of colours, for example $r_1^{r,b}$ and $r_2^{r,b}$

Fig. 22 Mean of cumulative regret over 50 trials. Compares situations where the teacher either does or does not use anaphora. The agent additionally either updates on teacher silence or does not, as in the previous experiment

three such rules. The agent's cumulative regret is shown in Fig. 20 with the results summarised in Table 4 and the difference in terminal regret visualised in Fig. 21. The paired t-test shows a significant improvement from exploiting the teacher's implied assent ($t = 5.42$, $p = 4.16e^{-7}$). This is supported by the fact that in about 75% of cases the agent which uses assent has a lower terminal regret as shown in Fig. 21.

The performance increase does come at the cost of an increase in the potential running time of the algorithm. We found that updating the model every time no correction was given and taking into account every possible rule caused some situations where the inference graph was too big and inference was infeasible (since exact inference in a Bayes Net is exponential). This is why we added the additional constraints as described in Sect. 5.2.4. With these changes, little discernible difference between inference time was found but there were still benefits to exploiting implied assent.

Assent benefits the agent by providing evidence that supports inferences as to which rules are *not* in the goal and it also provides additional positive exemplars of colours: e.g., if the agent believes $r_2^{r,b}$ is in the goal and a block o_1 is placed on something which is almost certainly *blue*, then silence implies o_1 is red. Exploiting silence as assent could also be used to learn concepts such as "gently". An utterance such as "no, put that down gently" is by default said in a context where the agent's action wasn't gentle. However, if a subsequent

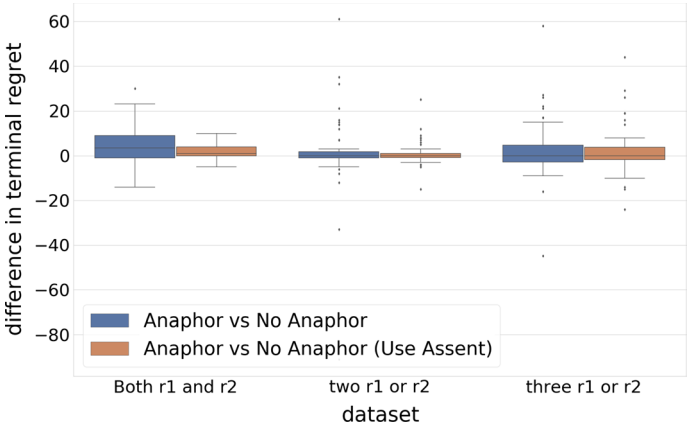


Fig. 23 This box plot shows the difference in terminal regret comparing agents which use anaphor and do not use anaphor (with our without using assent). Positive values mean the No Anaphor agent had higher terminal regret than the Anaphor agent

put action involving the same object is executed and there's no correction, then the agent might be able to infer that it's a positive example of "put gently".

6.3 Experiment 3

The third experiment tests the hypothesis that anaphora will lead to faster learning, thanks to the ways in which they can reduce overall ambiguity in the discourse (see Sect. 4.3 for discussion). We compare trials where the teacher uses anaphora against trials where she does not. We use the two types of learning agent from experiment 2. The results are shown in Fig. 22 and summarised in Table 4 with the difference in terminal regret visualised in Fig. 23. For the first trials, anaphora provides a very slight improvement, which is close to, but not quite, significant ($t = 2.4$, $p = 0.16$). However, in the third trial where two r_1 and r_2 rules share colours, the improvement is large and highly significant ($t = 4.57$, $p = 1.39e^{-5}$). This is supported by the evidence in Fig. 23. We see that the difference in terminal regret is higher in about 75% of runs for both the agents using Anaphora when we have both the r_1 and r_2 rule, but for the other two experiments the median difference lies around 0 implying the Anaphora did not consistently help. These results are empirical evidence that exploiting anaphora doesn't hurt, and indeed it can help in certain circumstances. The reason it is so helpful when both $r_1^{x,y}$ and $r_2^{x,y}$ are in the goal G is that by default the agent prefers explanations where only one of these rules is in the goal, due to the low prior. The inference that the current feedback not only corrects the latest action but also elaborates on a prior error, which the coherent use of anaphora enforces, helps provide additional information to override a faulty, defeasible prior.

We've also shown that anaphora can be easily integrated into our framework, which is important for ITL since people frequently use anaphoric expressions. However, anaphora will not reduce ambiguity in all contexts: utterances such as "put them on green blocks" involves the difficulty of interpreting what "them" refers to. We leave dealing with this type of anaphora to future work and refer readers to, for example Williams and Scheutz [64] for further discussion on this topic.

Table 5 Mean Terminal Regret for agents using either the full evidence or the simplified evidence for problems with r r_3 rules where the number of objects of allowed is n

Problem	Full	Simplified
$r = 1, n = 1$	3.34 ± 0.39	4.08 ± 0.71
$r = 1, n = 2$	3.66 ± 1.17	5.28 ± 0.85
$r = 1, n = 3$	3.92 ± 1.40	6.12 ± 2.17
$r = 2, n = 1$	6.28 ± 1.67	8.08 ± 2.41
$r = 2, n = 2$	7.14 ± 1.22	10.32 ± 1.79
$r = 2, n = 3$	7.74 ± 1.13	11.74 ± 2.67

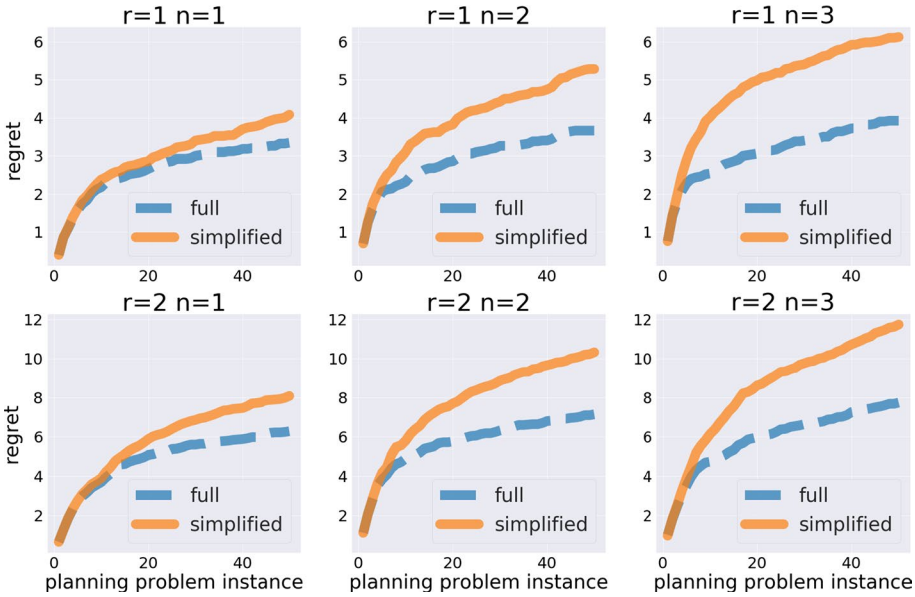


Fig. 24 Cumulative reward for experiments comparing agents which learn r_3 rules either using the full monotonic entailment or a simplified inference. The trials vary the number of rules, r, in the goals, between 1 and 2, and the maximum number of blocks of the constrained colour n between 1 and 3. The graphs show mean of regret over 50 trials

6.4 Experiment 4

In this paper we've exploited several semantic entailments of corrections to enhance task learning. However, corrections generate further entailments that require the agent to reason about the colours of blocks remaining on the table. For r_1 and r_2 rules, we have chosen to ignore these entailments, as we assumed the computational cost was too high compared to the benefit—exploiting such inferences makes the Bayes Net much less sparse, making the exponential complexity of exact inference a practical concern, not just a theoretical one. The introduction of r_3 rules allow us to test this assumption. These rules generate corrections with three monotonic semantic entailments that are associated with three levels of computational difficulty. The first is that the most recently placed object is of the relevant colour—thus adding one positive example of, say, red blocks. The cost of this is trivial, as there is only one non-zero probability option to consider. The second entailment is that

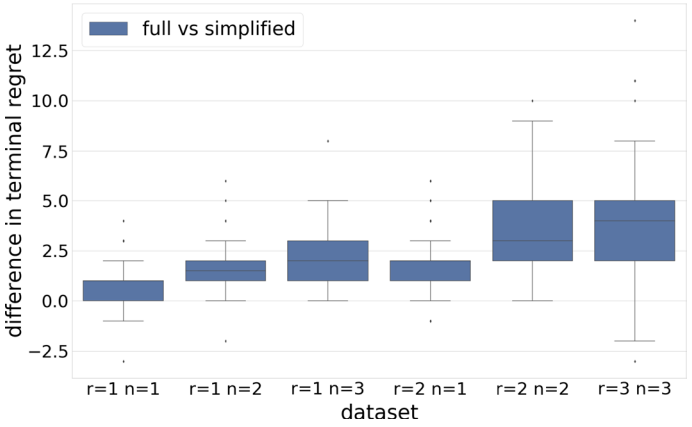


Fig. 25 This box plot compares the difference in terminal regret between agents using the full evidence or simple evidence on r_3 rules. A positive value means the agent using simple evidence had higher terminal regret than the agent using full evidence

Table 6 Results of t-tests comparing experiments for $r_3^{c,n}$ by either trying to estimate whether each of the blocks in the tower are the offending colour c or by just using the top block

r/n	1	2	3
1	$t = 4.52$, $p = 3.9e-5$	$t = 8.37$, $p = 5.22e-11$	$t = 9.70$, $p = 5.46e-13$
2	$t = 7.48$, $p = 1.22e-9$	$t = 9.01$, $p = 5.75e-12$	$t = 8.49$, $p = 3.45e-11$

The experiment was run with 1 or 2 rules of type $r_3^{c,n}$ where n varied between 1 and 3. For each experiment, the colours c for the rules are randomly selected from those that denote a relative large portion of the RGB spectrum

there are $n + 1$ blocks of the relevant colour in the tower which violated the rule. If the agent infers which blocks these are, then it can add an additional n, noisy, data points to train the grounding model (as we did with the Language Agent in experiment 1). However, the computational cost is that 2^t probability states must be evaluated, where t is the number of blocks in the tower. The final entailment is when the teacher tells the agent both r_3 and r_1 or r_2 were violated. In this case, as we mentioned in Sect. 4.2, the agent can attempt to infer whether the violated rule is r_1 or r_2 by reasoning about all blocks on the table, to count whether there is one more block of one colour than the other. However, the computational cost for this inference is steep, it adds an additional 2^{2b} probability states, where b is the number of blocks on the table.

We implement these three strategies. Initial experiments with the third entailment showed that the computational cost was too high: the agent would consistently run out of memory trying to evaluate roughly 2^{20} states. We call the remaining two strategies *simplified* for the first and *full* for the second. Figure 24 compares the performance of the simplified vs. full agents mean terminal regret summarised in Table 5. The number of $r_3^{c,n}$ rules is varied from varied from 1 to 2 and n is varied between 1 and 3. The *full* evidence provides a distinct benefit on cumulative regret for each type of trial, and Table 6 shows all these differences are significant. However, the benefit for $r_3^{c,n}$ is smaller than that for $n = 2$ and $n = 3$. This may be because the number of additional

data points that can be added to the grounding model depends on n: the higher the n, the more (noisy) data points a single correction provides. This is corroborated when looking at Fig. 25 which shows the difference in terminal regret between the two agents in each of the experiments where the pattern of full evidence being an improvement continues and where more complex problems gain higher advantage.

It is clear from these experiments that performing the more difficult inference can be highly valuable, but we have also seen that some inferences are too costly. Designers of similar systems will have to consider this trade off and select for themselves which entailments from linguistic signals to exploit, given their constraints. Additional computation time will lead to waiting time after feedback is given, which could cause human users to grow frustrated; however, they may become equally frustrated if they have to correct the agent for the same error several times. Ultimately all one can say is that there can be a marked difference in how quickly the agent learns, depending on which entailments are exploited.

7 Conclusion

In this paper, we exploited the semantics of coherent discourse to jointly learn three tasks via natural language interaction with a teacher: how to refine an agents domain model to include unforeseen concepts as and when they are discovered via neologisms in the interaction; how to ground the domain model's symbols to visual features; and how to infer a correct goal description, so as to construct valid sequential plans. We focused on a type of dialogue move that has so far been under utilised in ITL: namely, corrective feedback, in which the teacher expresses the nature of the mistake that the learner has made in its latest action. We showed that in spite of the teacher's natural language signals being semantically ambiguous and also featuring neologisms whose denotations are not a part of the agent's domain model, the agent can nevertheless exploit discourse coherence to learn how to compute the speaker's intended message, ground it via its visual percepts, and solve its planning problem. Our approach generates a graphical model 'on the fly' whenever there is observable evidence, with the dependencies and parameters determined by established semantic constraints on coherent discourse within the linguistics literature ([8, 29, 37], e.g.,). These models force the probability of incoherent interpretations of an utterance to be zero, thus ensuring the agent makes inferences consistent with the coherence of the dialogue. We have shown how to construct this graphical model for corrections, implied assent, and simple anaphoric expressions.

We evaluate our approach with four sets of experiments. We show that our agent outperforms a baseline which only learns from the teacher saying "no", even though her message is latent because the linguistic forms feature neologisms and semantic scope ambiguities. We also show that making use of the teacher's implied assent when she is silent and her use of anaphoric expressions improves how quickly the agent learns. Finally, we explore the trade-off between exploiting a monotonic semantic consequence of a coherence relation on the one hand, and the size of the search space that's needed to verify that this semantic consequence is satisfied by the current domain state on the other. For some entailments, it is completely impractical to learn from them without running out of memory: the number of dependencies that must be added to the Bayes Net to reflect certain monotonic entailments of the message creates a situation where exponential exact inference is a practical impediment, not just a theoretical one. For

other entailments, however, exploiting them can improve the agent's performance with respect to cumulative regret, but the time required to perform the necessary inferences is longer and may lead to the system becoming frustrating for human teachers. Overall, exploiting the semantics of discourse coherence is useful, indeed critical; but the system designer shouldn't feel compelled to exploit every semantic entailment as evidence: one needs to find an optimal trade off between data efficiency during learning vs. processing time and memory.

While we've shown a promising approach for exploiting discourse coherence to ITL, there are still several questions that need to be addressed before the approach is ready to be deployed with real humans. Firstly, we show our approach works for grounding colours; however, any real scenario will involve more challenging grounding problems, which require more sophisticated grounding models. Our grounding models are in principle replaceable with any state of the art technique, so long as it outputs a probability distribution and can support incremental updates and noisy training data. But the effects on performance are clearly an empirical matter. We also consider only very specific types of planning problems. For instance, the actions we consider are reversible, i.e. we can return to the state before the action was executed by using some other action, such as *unstack*. Our learning models don't rely on the actions being reversible, but the current implementation does, which means it is an empirical question as to whether the method would still work in other conditions. As we mentioned in Sect. 3, planning problems where actions aren't reversible are generally more computationally complex to solve than planning problems where they are, and we have yet to investigate how the models we've developed here will scale when the planning problems to be solved are more complex.

Secondly, we focused on a simple dialogue strategy: when a rule (or combination of rules) in the goal G is violated, the teacher corrects the agent by expressing, albeit in a potentially ambiguous way, which part of G is violated. But while this strategy is simple to express, listing all conditions under which a rule (or combination of rules) is violated is non-trivial, and it may not scale easily to increase the type of constraints we cover. An ideal solution would be to use automated theorem proving, model builders and model checkers to automatically generate the valid hypothesis space given an arbitrary new constraint, something we intend to investigate further in future work. Thirdly, we assumed that the teacher and learner share common knowledge about their dialogue strategy. But in natural human dialogue, the coherence relation that connects a speaker's utterance to its context isn't observable [8, 30]. So in a general system, discourse parsing will be necessary (e.g., [1, 46]), and in contrast to the assumptions in this paper its estimates of coherence relations may be wrong. Additionally, when it comes to exploiting the teacher's silence as assent, there is a significant challenge to overcome in face to face communication of actually detecting when the teacher is implying assent and when they simply aren't paying attention or perhaps are slow to react or are quiet for any other reason. We skirt around this by acting in a virtual environment, giving the teacher time to react to each action. But in order to deploy our system in an embodied setting with human teachers, the model would have to be updated to deal with these problems.

Finally, our model assumes the teacher is infallible: always coherent, sincere and competent. In future work, we plan to address uncertainty about the coherence relations and imperfect teachers by refining the graphical models generated from observable evidence and adjusting their prior probabilities.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long

as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Afantenos, S. D., Kow, E., Asher, N. & Perret, J. (2015). Discourse parsing for multi-party chat dialogues. In L. Màrquez, C. Callison-Burch, J. Su., D. Pighin & Y. Marton (Eds.), *Proceedings of the 2015 conference on empirical methods in natural language processing, EMNLP 2015, Lisbon, Portugal*, September 17-21, 2015, The Association for Computational Linguistics (pp. 928–937) https://doi.org/10.18653/v1/d15-1109.
2. Al-Omari, M., Duckworth, P., Hawasly, M., Hogg, D. C. & Cohn, A. G. (2017). Natural language grounding and grammar induction for robotic manipulation commands. In M. Bansal, C. Matuszek, J. Andreas, Y. Artzi & Y. Bisk (Eds.), *Proceedings of the first workshop on language grounding for robotics, RoboNLP@ACL 2017*, Vancouver, Canada, August 3, 2017, Association for Computational Linguistics (pp. 35–43) https://doi.org/10.18653/v1/w17-2805.
3. Anderson, P., Wu, Q., Teney, D., Bruce, J., Johnson, M., Sünderhauf, N., Reid, I. D., Gould, S. & van den Hengel, A. (2017). Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *2018 IEEE/CVF conference on computer vision and pattern recognition* (pp. 3674–3683).
4. Appelgren, M. & Lascarides, A. (2019a). Coherence, symbol grounding and interactive task learning. In *Proceedings of the 23rd workshop on the semantics and pragmatics of dialogue: full papers, SEMDIAL*, London, United Kingdom. http://semdial.org/anthology/Z19-Appelgren_semdial_0004.pdf.
5. Appelgren, M. & Lascarides, A. (2019b). Learning plans by acquiring grounded linguistic meanings from corrections. In *Proceedings of the 18th inter- national conference on autonomous agents and multiagent systems (AAMAS 2019)* Montreal, Canada, May 13–17, 2019, IFAAMAS (p. 9).
6. Argall, B., Chernova, S., Veloso, M., & Browning, B. (2009). A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, *57*, 469–483.
7. Asher, N., & Lascarides, A. (1998). The semantics and pragmatics of presupposition. *Journal of Semantics*, *15*(2), 239–299.
8. Asher, N., Asher, N. M., & Lascarides, A. (2003). *Logics of conversation*. Cambridge: Cambridge University Press.
9. Bastianelli, E., Bloisi, D. D., Capobianco, R., Gemignani, G., Iocchi, L. & Nardi, D. (2013). Knowledge representation for robots through human-robot interaction. arXiv:abs/1307.7351.
10. Bird, S., & Liberman, M. (2000). A formal framework for linguistic annotation (revised version). CoRR,. arXiv:cs.CL/0010033.
11. Bylander, T. (1994). The computational complexity of propositional strips planning. *Artificial Intelligence*, *69*, 165–204.
12. Calhoun, S. (2006). Information structure and the prosodic structure of english: A probabilistic relationship. PhD thesis, University of Edinburgh.
13. Copestake, A., Flickinger, D. & Sag, I. A. (1999). Minimal recursion semantics: An introduction.
14. Copestake, A. A. & Flickinger, D. (2000). An open source grammar development environment and broad-coverage english grammar using HPSG. In *Proceedings of the second international conference on language resources and evaluation, LREC 2000*, 31 May–June 2, 2000, Athens, Greece, European Language Resources Association. http://www.lrec-conf.org/proceedings/lrec2000/html/summary/371.htm.
15. Coradeschi, S., & Saffiotti, A. (2003). An introduction to the anchoring problem. *Robotics and Autonomous Systems*, *43*, 85–96.
16. Coradeschi, S., Loutfi, A., & Wrede, B. (2013). A short review of symbol grounding in robotic and intelligent systems. *KI - Künstliche Intelligenz*, *27*, 129–136.
17. Dobnik, S. & de Graaf, E. (2017a). Kille: A framework for situated agents for learning language through interaction. In *NODALIDA*.
18. Dobnik, S. & de Graaf, E. (2017b). Kille: Learning grounded language through interaction. In *FADLI@ESSLLI*.

19. Erdem, E., Aker, E., & Patoglu, V. (2012). Answer set programming for collaborative housekeeping robotics: Representation, reasoning, and execution. *Intelligent Service Robotics*, *5*, 275–291.
20. Erdem, E., Gelfond, M., & Leone, N. (2016). Applications of answer set programming. *AI Magazine*, *37*, 53–68.
21. Fang, R., Liu, C., She, L. & Chai, J. Y. (2013). Towards situated dialogue: Revisiting referring expression generation. In *EMNLP*.
22. Forbes, M., Rao, R. P. N., Zettlemoyer, L. & Cakmak, M. (2015). Robot programming by demonstration with situated spatial language understanding. In *IEEE international conference on robotics and automation, ICRA 2015*, Seattle, WA, USA, 26-30 May, 2015 (pp. 2014–2020). https://doi.org/10.1109/ICRA.2015.7139462.
23. Gemignani, G., Capobianco, R., Bastianelli, E., Bloisi, D. D., Iocchi, L., & Nardi, D. (2016). Living with robots: Interactive environmental knowledge acquisition. *Robotics and Autonomous Systems*, *78*, 1–16.
24. Giménez, O. & Jonsson, A. (2008). The complexity of planning problems with simple causal graphs. arXiv:abs/1111.0056.
25. Grice, H. P. (1975). Logic and conversation. *1975*, 41–58.
26. Gupta, N., & Nau, D. S. (1992). On the complexity of blocks-world planning. *Artificial Intelligence*, *56*, 223–254.
27. Harnad, S. (1990). The symbol grounding problem. *CoRR*,. arXiv:cs.AI/9906002.
28. Hobbs, J. (1979). Coherence and coreference. *Cognitive Science*, *3*(1), 67–90.
29. Hobbs, J. R. (1985). On the coherence and structure of discourse. Tech. Rep. csli-85-37, Center for the Study of Language and Information, Stanford University, Stanford.
30. Hobbs, J. R., Stickel, M. E., Appelt, D. E., & Martin, P. A. (1993). Interpretation as abduction. *Artificial Intelligence*, *63*(1–2), 69–142. https://doi.org/10.1016/0004-3702(93)90015-4.
31. Hoffmann, J. (2003). The Metric-FF planning system: Translating "ignoring delete lists" to numeric state variables 20:291–341
32. Hoffmann, J., & Nebel, B. (2001). The FF planning system: Fast plan generation through heuristic search. *14*, 253–302.
33. Hristov, Y., Penkov, S., Lascarides, A. & Ramamoorthy, S. (2017). Grounding symbols in multi-modal instructions. In *Proceedings of the first workshop on language grounding for robotics, RoboNLP@ACL 2017*, Vancouver, Canada, August 3, 2017 (pp. 49–57). https://aclanthology.info/papers/W17-2807/w17-2807.
34. Hristov, Y., Lascarides, A. & Ramamoorthy, S. (2018). Interpretable latent spaces for learning from demonstration.
35. Hunter, J., Asher, N., & Lascarides, A. (2018). A formal semantics for situated conversation. *Semantics and Pragmatics*,. https://doi.org/10.3765/sp.11.10.
36. Karamcheti, S., Williams, E. C., Arumugam, D., Rhee, M., Gopalan, N., Wong, L. L. S. & Tellex, S. (2017). A tale of two draggns: A hybrid approach for interpreting action-oriented and goal-oriented instructions. In *Proceedings of the first workshop on language grounding for robotics, RoboNLP@ACL 2017*, Vancouver, Canada, August 3, 2017 (pp. 67–75). https://aclanthology.info/papers/W17-2809/w17-2809.
37. Kehler, A. (2002). Coherence, Reference and the Theory of Grammar. csli Publications, Cambridge University Press.
38. Kirk, J. R. & Laird, J. E. (2019). Learning hierarchical symbolic representations to support interactive task learning and knowledge transfer. In *Proceedings of the twenty-eighth international joint conference on artificial intelligence, IJCAI 2019*, Macao, China, August 10–16, 2019 (pp. 6095–6102) https://doi.org/10.24963/ijcai.2019/844.
39. Knox, W. B. & Stone, P. (2009). Interactively shaping agents via human reinforcement: The TAMER framework. In *Proceedings of the 5th international conference on knowledge capture (K-CAP 2009)*, September 1–4, 2009, Redondo Beach, California, USA (pp. 9–16) https://doi.org/10.1145/1597735.1597738.
40. Kollar, T., Tellex, S., Roy, D. & Roy, N. (2010). Toward understanding natural language directions. In *Proceedings of the 5th ACM/IEEE international conference on human robot interaction, HRI 2010*, Osaka, Japan, March 2–5, 2010 (pp. 259–266) https://doi.org/10.1145/1734454.1734553.
41. Laird, J. E., Gluck, K. A., Anderson, J. R., Forbus, K. D., Jenkins, O. C., Lebiere, C., et al. (2017). Interactive task learning. *IEEE Intelligent Systems*, *32*, 6–21.
42. Larsson, S. (2018). Grounding as a side-effect of grounding. *Topics in Cognitive Science*, *10*(2), 389–408.
43. Lascarides, A. & Asher, N. (2003). Imperatives in dialogue. *Pragmatics and Beyond New Series* pp 1–24.

44. Lascarides, A., & Asher, N. (2009). Agreement, disputes and commitments in dialogue. *Journal of Semantics*, *26*(2), 109–158. https://doi.org/10.1093/jos/ffn013.
45. Lauria, S., Bugmann, G., Kyriacou, T., & Klein, E. (2002). Mobile robot programming using natural language. *Robotics and Autonomous Systems*, *38*(3–4), 171–181. https://doi.org/10.1016/S0921-8890(02)00166-5.
46. Liu, Y., & Lapata, M. (2018). Learning structured text representations. *Transactions of the Association for Computational Linguistics*, *6*, 63–75. https://doi.org/10.1162/tacl_a_00005.
47. Mann, W. C., & Thompson, S. A. (1986). Rhetorical structure theory: Description and construction of text structures. In Natural Language (Ed.), Kempen, G (pp. 279–300). New Results in Artificial Intelligence: Generation.
48. Matuszek, C. (2018). Grounded language learning: Where robotics and nlp meet. In *IJCAI*.
49. Naim, I. (2015). Unsupervised alignment of natural language with video.
50. Nicolescu, M. N. & Mataric, M. J. (2001). Experience-based representation construction: learning from human and robot teachers. In *IEEE/RSJ international conference on intelligent robots and systems, IROS 2001: Expanding the societal role of robotics in the the next millennium*, Maui, HI, USA, October 29–November 3, 2001 (pp. 740–745) https://doi.org/10.1109/IROS.2001.976257.
51. Nicolescu, M. N. & Mataric, M. J. (2003). Natural methods for robot task learning: Instructive demonstrations, generalization and practice. In *The second international joint conference on autonomous agents & multiagent systems, AAMAS 2003*, July 14–18, 2003, Melbourne, Victoria, Australia, Proceedings (pp. 241–248) https://doi.org/10.1145/860575.860614.
52. Rooth, M. (1992). A theory of focus interpretation. *Natural Language Semantics*, *1*(1), 75–116.
53. Scheutz, M., Krause, E. A., Oosterveld, B., Frasca, T. M. & Platt, R. (2017). Spoken instruction-based one-shot object and action learning in a cognitive robotic architecture. In *AAMAS*.
54. She, L. & Chai, J. Y. (2017). Interactive learning of grounded verb semantics towards human-robot communication. In *ACL*.
55. She, L., Yang, S., Cheng, Y., Jia, Y., Chai, J. Y. & Xi, N. (2014). Back to the blocks world: Learning new actions through situated human-robot dialogue. In *SIGDIAL Conference*.
56. Silberer, C. & Lapata, M. (2014). Learning grounded meaning representations with autoencoders. In *ACL*.
57. Socher, R., Karpathy, A., Le, Q. V., Manning, C. D., & Ng, A. Y. (2014). Grounded compositional semantics for finding and describing images with sentences. *TACL*, *2*, 207–218.
58. Stryjewski, L. (2010). 40 years of boxplots.
59. Tellex, S., Knepper, R. A., Li, A., Rus, D. & Roy, N. (2014). Asking for help using inverse semantics. In *Robotics: Science and systems X*, University of California, Berkeley, USA, July 12–16, 2014. http://www.roboticsproceedings.org/rss10/p24.html.
60. Thomason, J., Sinapov, J., Svetlik, M., Stone, P. & Mooney, R. J. (2016). Learning multi-modal grounded linguistic semantics by playing "i spy". In *IJCAI*.
61. van der Sandt, R. (1992). Presupposition projection as anaphora resolution. *Journal of Semantics*, *9*(4), 333–377.
62. Vinyals, O., Blundell, C., Lillicrap, T., kavukcuoglu, K. & Wierstra, D. (2016). Matching networks for one shot learning. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, R. Garnett (Eds.), *Advances in neural information processing systems 29, Curran Associates, Inc.* (pp. 3630–3638) http://papers.nips.cc/paper/6385-matching-networks-for-one-shot-learning.pdf.
63. Wang, S. I., Liang, P. & Manning, C. D. (2016). Learning language games through interaction. In *Proceedings of the 54th annual meeting of the association for computational linguistics, ACL 2016*, August 7–12, 2016, Berlin, Germany, Volume 1: Long Papers http://aclweb.org/anthology/P/P16/P16-1224.pdf.
64. Williams, T. & Scheutz, M. (2018). Reference in robotics: A givenness hierarchy theoretic approach. The Oxford handbook of reference.
65. Yu, H., Zhang, H., & Xu, W. (2018). Interactive grounded language acquisition and generalization in a 2d world. CoRR,. arXiv:abs/1802.01433.
66. Yu, Y., Eshghi, A. & Lemon, O. (2017). Learning how to learn: An adaptive dialogue agent for incrementally learning visually grounded word meanings. In *Proceedings of the first workshop on language grounding for robotics, RoboNLP@ACL 2017*, Vancouver, Canada, August 3, 2017 (pp. 10–19) https://aclanthology.info/papers/W17-2802/w17-2802.
67. Zettlemoyer, L. S. & Collins, M. (2007). Online learning of relaxed ccg grammars for parsing to logical form. In *EMNLP-CoNLL*.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.